DOCUMENT RESUME

ED 178 067 IR 007 809

AUTHOR	Prather, Ronald E., Ed.
TITLE	Proceedings of 1978 Conference on Computers in the Undergraduate Curricula: CCUC9 (9th, Denver, CO, June 12-14, 1978).
PUB DATE	Jun 78
NOTE	393p.; For related documents, see IR 007 810 and ED 156 160
AVAILABLE FROM	111 Weeg Computing Center, University of Iowa, Iowa City, Iowa 52242 ($10.00 per copy)
EDRS PRICE	MF01 Plus Postage. PC Not Available from EDRS.
DESCRIPTORS	Business Education; *Computer Assisted Instruction; *Computer Managed Instruction; Computer Programs; *Computer Science; Humanities Instruction; Mathematical Models; Mathematics Instruction; Programed Instruction; Social Sciences; Statistics

ABSTRACT

The 48 papers in this conference report are concerned with the use of computers in undergraduate curricula in various educational settings. Articles are split into general topic headings, including Computer Augmented Video Education (CAVE), Mathematics, Simulation, Testing, Humanities, Social Science, Computer Assisted Instruction/Computer Managed Instruction, Computer Science, Business, Science, Statistics, and Computer Services. (JEG)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED178067

U S DEPARTMENT OF HEALTH.
EDUCATION & WELFARE
NATIONAL INSTITUTE OF
EDUCATION

THIS DOCUMENT HAS BEEN REPRODUCED EXACTLY AS RECEIVED FROM THE PERSON OR ORGANIZATION ORIGINATING IT. POINTS OF VIEW OR OPINIONS STATED DO NOT NECESSARILY REPRESENT OFFICIAL NATIONAL INSTITUTE OF EDUCATION POSITION OR POLICY

PROCEEDINGS OF

1978

CONFERENCE ON COMPUTERS
IN THE
UNDERGRADUATE CURRICULA

CCUC9

Edited by

RONALD E. PRATHER

Hosted by

UNIVERSITY OF DENVER

JUNE 12-13-14, 1978

"PERMISSION TO REPRODUCE THIS MATERIAL IN MICROFICHE ONLY HAS BEEN GRANTED BY

Ted Sjoerdsma

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

IR007809

STEERING COMMITTEE

Gerald L. Engel
Old Dominion University

John W. Hamblen
University of Missouri at Rolla

Harry Hedges
Michigan State University

James A. Higgins
Digital Equipment Corporation

Glenn R. Ingram
U.S. Energy Research and Development Administration

Theodore J. Sjoerdsma
University of Iowa

David L. Stonehill
State University of New York at Binghamton

Alfred Bork
University of California at Irvine

William S. Dorn
University of Denver

LOCAL COORDINATING COMMITTEE

Chairman
William S. Dorn

Coordinator
Rick Yuskaitis

Proceedings
Ronald E. Prather

Program
Judith D. Schlesinger

Housing and Registration
Herbert H. Smith
Shelby Worley, Tom Williams

Food Services
Ski Adamczyk

Transportation
Joel S. Cohen

Exhibits
F. B. Lucas

Audio-Visual Equipment
Rick Yuskaitis

Tour of Coors Brewery
A. E. Ritter

Sunday Evening Mixer
H. J. Greenberg
Becky Roberts

DU Conference Office
Anne Davis

Message Center
Anita Johnson
Donald Weiss

The Ninth Conference on Computers in the Undergraduate Curricula wishes to thank the following people for their contribution of effort, time, and knowledge as referees of the papers submitted for presentation.

Ronald E. Anderson, University of Minnesota
Richard H. Austing, University of Maryland
Alfred Bork, University of California at Irvine
G. Robert Boynton, University of Iowa
W. G. Bulgren, University of Kansas
Phillip L. Carter, Michigan State University
Ronald Collins, Eastern Michigan University
Joyce Corrington, Xavier University of Louisiana
M. K. Cueman, Virginia Institute of Marine Science
William S. Dorn, University of Denver
B. G. Dunn, Fairmont State College
G. L. Engel, Old Dominion University
Selby Evans, Texas Christian University
Armando J. Galfo, College of William and Mary
Sheldon P. Gordon, Suffolk County Community College
Carl Hacker, University of Texas Health Service Center
John W. Hamblen, University of Missouri at Rolla
Harry Hedges, Michigan State University
Grace Hertlein, California State University, Chico
James A. Higgins, Digital Equipment Corporation
A. A. J. Hoffman, Texas Christian University
Glenn Ingram, U.S. Energy Research and Development Administration
Lawrence Jehn, University of Dayton
Vincent H. Jones, Southern University
Joyce C. Little, Community College of Baltimore
Donald H. McClain, University of Iowa
Donald E. McLaughlin, Augustana College
Andrew Molnar, National Science Foundation
Robert D. Montgomery, North Carolina Central University
Linda Petty, Hampton Institute
Ernest Pope, Palo Alto Unified School District
W. W. Porterfield, Hampden-Sydney College
C. A. Quarles, Texas Christian University
Joseph Raben, Queens College of the City University of New York
Ottis Rechard, University of Denver
Hassell Simpson, Hampden-Sydney College
Ted Sjoerdsma, University of Iowa
David Smith, Duke University
Phillip F. Spelt, Wabash College
Ronald Stiff, University of Baltimore
David Stonehill, State University of New York at Binghamton
John L. VanIwaarden, Hope College
Rufus F. Walker, Jr., Louisiana State University Medical Center
T. C. Willoughby, Ball State University
Frank Wojcik, Virginia Institute of Marine Science
Allen D. Ziebur, State University of New York at Binghamton

We also thank the authors for keeping the publication on schedule.

Ronald E. Prather

TABLE OF CONTENTS

CAVE I

Computer Augmented Video Education: An Overview 1
Richard A. Pollak

Computer Augmented Video Education (CAVE) in Chemistry: Laboratory Exercises 5
Frank J. Gomba

Computer Augmented Video Education (CAVE) in Troublesome Areas of Chemistry 8
John V. Prestia

MATHEMATICS I

The Effects of CAI on Achievement and Attitude in the Freshman Survey Mathematics Curriculum 16
Leo Edwards, Jr.

A General Problem Solver for Mathematics Courses 23
Roger C. Barr, Thomas M. Gallie, Jr., Margaret J. Hodel, Richard E. Hodel, Francis J. Murray, David A. Smith

Interweaving Mathematics and Computers for Liberal Arts Students 36
Kenneth John Preskenis

CAVE II

Computer Augmented Video Education (CAVE) in Economics 42
Rae Jean B. Goodman

CAVE - Computer Augmented Video Education for Basic Physics 47
Billie J. Graham

MATHEMATICS II

Computer Supplemented Calculus Instruction 53
Robert A. McCoy

MATHDOC: A Control System for Computer Assisted Calculus 57
Mark J. Christensen

SIMULATION I

Tigers and Rabbits: Curricular Uses of a System Dynamics Model of a Simple Idealized Ecosystem 68
Priscilla W. Laws

A Lens and Mirror Simulation Program 84
Robert L. Siddon

TESTING

The Use of Peerrate: A Computerized System of Student Participation in Grading 87
Anthony D. Lutkus

PROBGEN: An Interactive Computer Program for Faculty Use in Creating and Generating Individualized Nonidentical Problem Sets and Tests 100
Ronald W. Collins, Steven J. Duff

HUMANITIES

Generative CAI in Aristotelian Logic 117
Walter Maner

Generating English Programs at a Small College 127
Gayle Byerly

An Undergraduate Course in Computer-Aided Musical Composition Utilizing Time-Shared BASIC 133
Clark Dehne, Bert A. Anderson

A New Approach to Computer Assisted Instruction in Music Theory 142
Richard L. Hamilton, Dan W. Scott

SOCIAL SCIENCE

A Strategy for Using Computer Programs in Teaching Cartography 151
James W. Cerny

A Computer-Integrated Course in Research Methods 156
Arthur St. George

Computer Assisted Homework and Teaching Effectiveness: A Case Study from Introductory Sociology 168
George H. Conklin

Development of Computer-linked Education in Research Psychology 180
John L. Horn, Jack McArdle

CAI/CMI

QUATE: A Standardized Procedure for Describing Structured CAI Materials 189
Doug Aims

The Mini-Computer as an Interactive Time-Share Instructional System for the Classroom 196
David L. Jelden

A Graphic Design Tool to Aid in Authoring Student-Computer Dialogs 201
Mark Geisert, Alfred Bork

COMPUTER SCIENCE I

The Computer, The Individual and Society: A Course for Secondary Teachers 206
Stanley F. Robinson

Undergraduate Programs to Increase Instructional Computing in Schools 212
J. Richard Dennis

What Makes a Computer-Literate College? 219
Beverly Hunter

BUSINESS

Computer-Assisted Forecasting of Economic Aggregates 227
Ronald A. Krieger

Making Large Classes Work in Computing for Business 236
Thomas Rigo, John M. Gersting

Development and Implementation of an Interactive Accounting System 243
Louis Geller

Cash Flow: An Interactive Approach Designed to Assist in Student Understanding of the Principles of Cash Management 249
Richard T. Nyerges, Harry L. Reif

SCIENCE

Musical Acoustics Experiments with a Laboratory-Based Computer 260
G. T. Wood

Computerized Solar Heating Analysis 268
Richard A. Blade, Michael H. Scott

The Design and Implementation of Computer Assisted Instructional Modules in Engineering Physics 273
Grayson H. Walker, Homer L. Utley

Population Dynamics with a Computer 278
Robert P. Banaugh

COMPUTER SCIENCE II

Minicomputer Systems as a Vehicle for Teaching Computer and Operating Systems 289
Emil C. Neu

Introductory Programming Courses Without Lectures 297
George Nagy

Introduction to Computing - A Modular Approach 304
Phyllis A. Kent, Marguerite A. Boisvert

STATISTICS

Computer Aided Instruction in Business Statistics 309
Chiou-Nan Yeh, John A. Willhardt

The Computer in an Introductory Statistics Course 314
K. Rappaport, T. Stafford

Interactive Computing in Teaching Statistical Concepts 324
Z. A. Karian, A. Sterrett

COMPUTER SERVICES

The Effect of CAUSE on Instructional Computing at a Liberal Arts College 334
David D. Dexter

Development of a Computer Oriented Program in Science and Mathematics 344
Sheldon P. Gordon

Computer Resource Center - A Consolidation of Instructional Department Terminals 352
Victor Langer

SIMULATION II

A Computer Simulation for Teaching Client Referral Skills 356
William Abernathy, Elizabeth McBride

The Application of Simulation Techniques to Enhance Student Understanding of Commercial Bank Management 364
Alan K. Reichert, N. J. Greenberg

Interactive Computer Models for Introducing Simulation Techniques 379
William C. Nylin, Jr., David R. Read

COMPUTER AUGMENTED VIDEO EDUCATION: AN OVERVIEW

by

Richard A. Pollak
Associate Director for
Educational Development
Academic Computing Center
U. S. Naval Academy
Annapolis, Maryland 21402

The Computer Augmented Video Education (CAVE) Project at the Naval Academy was designed as a pilot study to combine the unique instructional advantages of educational television and academic computing systems. That is, the interactive powers of the computer have been interfaced with the audio-pictorial capabilities of education television, forming one powerful instructional resource. The components of this resource already exist at the Naval Academy's Educational Resources Center and the Academic Computing Center.

The Academy has a sophisticated educational television network which includes 12 closed-circuit TV channels and one of the most up-to-date TV studios. An outgrowth of this facility is a specialized Audio-Visual Room designed to allow students better control of their learning processes. The room comprises 60 carrels, 16 of which have built-in TV monitors connected to individual, 3/4-inch video playback cassette units.

The Naval Academy's Academic Computing Center is a separate organization, with the mission of supporting the faculty in terms of: providing digital computer service, technical support and consultation, instruction, and dissemination of innovative educational computer techniques. It operates a Honeywell-635 computer under the Dartmouth Time-Sharing System. There are about 256 terminal access ports into the computer, and presently there are over 300 interactive terminals that can access the computer system from various locations in nine separate buildings. The major underlying guideline in interfacing these two separate forms of educational technology into computer-augmented video instruction was to use existing, commercially available hardware at the lowest cost. Some of the main constraints put on the interface were that it had to be:

1. Small enough to fit into the physical area of the existing built-in video playback carrels.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

2. Directly interfaced with existing educational TV equipment without duplication of resources (i.e., the computer terminal selected should come as a key-pad without its own TV monitor).
3. Equipped with a key-pad having its own computer interface (EIA/RS232C) and composite video output interface (EIA/RS170).
4. Able to switch from 40 to 80 characters per line.

The illustration on the last page shows one of the four carrels set up to be used in the CAVE Project. With existing video cassette players included, the cost of setting up each carrel came to about $1,500.

In interfacing the computer with the video cassette player, the underlying idea was to make the interface simple for the student to operate, while keeping the costs and modifications to a minimum. It was determined that a two-button switchover would be the easiest. These switches already existed on the monitor we used. Note the lower right corner of the TV monitor in the illustration. By depressing the computer "transfer" button, the video cassette unit would be put on hold, and the video output would come from the computer terminal. This was a simple modification handled by ETV personnel. By depressing the video "transfer" button, the video output would come from the cassette player.

The pilot study was designed in three phases. The first phase took place between October 1976 and May 1977, during which time we discussed investigation into the following: topics which could be best presented through the CAVE format, hardware capabilities, video and computer software capabilities. The final outcome of this phase was that each CAVE member was to develop one program in his own discipline. Using curriculum designed specifically for the project, the programs were to be designed to test the instructional capabilities of the concept as well as the actual mechanics of interfacing the computer with the video cassette player. The second phase took place during the summer of 1977 at which time each CAVE member developed two to six programs. Each member was responsible for both the computer design of instruction and the TV presentation. These individuals were supported by technical advisers from the Academic Computing Center and the Educational Resources Center. The third phase, which is in progress now during the Fall Semester of 1977, is designed to evaluate formally the programs developed as well as the hardware and system software. Individuals were also encouraged to develop more courseware.

Two major types of evaluation will be performed during this project. First, the hardware and software systems will be analyzed to determine if this is the best equipment for interfacing the two educational resources. Secondly, a formative scheme of evaluation will be conducted on the effectiveness of instructional resources. "Formative evaluation is the use of systematic evaluation in the process of curriculum construction, teaching, and learning for the purpose of improving any of these processes. This process involves the collection of appropriate evidence during the construction and trying out a new curriculum in such a way that revisions of the curriculum can be based on this evidence."*

Evaluation will be in a form to answer the following:

1. Does the instructional program represent a true implementation of the objectives?
2. Does the instructional program, when developed and put into operation, achieve the desired outcomes?

---

* Bloom, Hastings, and Madaus. Handbook on Formative and Summative Evaluation of Student Learning. New York: McGraw-Hill, 1971, 2.

The following are foreseeable benefits of using the one combined resource:

1. Visual learning can be experienced while the student takes an active role in the learning process.
2. Remedial and make-up instruction can be tailored to individual student needs.
3. Instruction can be presented that may be otherwise impractical, time-consuming, or too involved for classroom presentation.
4. Theory can be presented pictorially while the interface to the computer can be used to "operationalize" the concepts into concrete examples.
5. Records can be kept that allow both the student and the instructor to know instantaneously the results of the student's learning experience.

In summary, the project is still being evaluated, but at this time it seems to be successful. The three main objectives, as follows, are well on their way to being accomplished:

1. To combine the inherent advantages of two separate forms of educational technology (computer/video) into one powerful instructional resource.
2. To create a computer-video interface at a low cost.
3. To expand the scope of benefit derived from an already existing A-V room and computer support of instructional systems.

Furthermore, throughout the pilot project, programs have been developed which use the full capabilities of both educational television and the computer. These programs use various modes of presentation including tutorial, laboratory preparation, drill/practice, and simulations. The project is multi-disciplinary and presently represented by chemistry, physics, and economics. Each member has designed curriculum in his/her own area of expertise, using various modes of preparation.

Specific examples elaborating on how CAVE is being used in actual instruction are in the following papers which also appear in this proceeding:

CAVE in Chemistry: Laboratory Exercises
CAVE in Economics
CAVE for Basic Physics
CAVE in Troublesome Areas of Chemistry

CAVE members are:

F. J. Gomba, Chemistry Department
R. J. B. Goodman, Economics Department
B. J. Graham, Physics Department
R. A. Pollak, Associate Director for Educational Development, Academic Computing Center
J. V. Prestia, Chemistry Department
M. B. Sousa, (Capt., USN), Assistant Dean for Educational Resources and Director of Computer Services
J. C. Thompson, Director, Educational Resources Center

Appreciation goes to all the "cave dwellers" for many hours of dedicated work that made the project possible. Special recognition is due James W. Schwab, James A. Kiser, and Al Wiess whose technical support and many hours of hard work made the project successful.

COMPUTER AUGMENTED VIDEO EDUCATION (CAVE) IN CHEMISTRY: LABORATORY EXERCISES

by

Frank J. Gomba
Chemistry Department
U. S. Naval Academy
Annapolis, Maryland 21402

The teaching business is somewhat like running an eating establishment. By-passing the apparent similarities that could be given for the physical plants, we could emphasize the bill of fare. Some students, as do the customers, like everything served to them. Of course, the quality of preparation may vary; and some students will take to the counter to meet the teacher part of the way. Perhaps most will want "the best table in the classroom" and wait for everything to come their way. And then we have those who like to pick up some things on their own in the buffet line.

Teaching a discipline has to be compared to a single entree. Just as the main dish may be served hot or cold -- with small or large portions, attention to detail, special wines and side dishes to enhance it -- so all these qualities may be incorporated into teaching a particular subject. Instructors want to make the subject palatable and have their pupils coming back for "seconds." Perhaps a "spoonful of sugar will help the medicine go down" for those who don't particularly like the dish or are afraid to try it. And a good dish may become better with more attention. It could even help overcome the old complaint: "not that again."

And so, we took a side dish of laboratory exercises in the main course of chemistry and tried to make it better. Laboratory experiments are the "fun part" of a chemistry course. You get "hands-on" experience and the good feeling of having things come out right. To help make the experience come out even better, we chose a couple of laboratory experiments to do via the CAVE project (Computer Augmented Video Education).

The project had a three-fold purpose:

1. To provide pre-laboratory instruction, emphasizing correct laboratory procedure and data collection via video programs.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

2. To provide drill on data treatment, both visually and by computer presentation.

3. To provide realistic, but randomly created, data that could be collected during an experiment by means of computer programs. If a student missed an experiment, this could be used along with the video segments to provide laboratory experience in a remote manner.

Although the video program and computer program could be used separately, the two programs augmented each other for a more complete teaching experience. In this recommended method of use (TV and computer together), the same television receiver is used for the video cassette program and for the output of the computer.

In this initial CAVE laboratory project, the two experiments were Stoichiometry of Potassium Chlorate and Spectrophotometry (visible).

The Stoichiometry of Potassium Chlorate involved detailed video programming showing close-ups of the laboratory apparatus, its proper set-up and usage, as well as collection of required data and data reduction. Students using the video program could see exactly how the apparatus was set up and how it was used. No written word or oral instructions could convey this as effectively and quickly. Data reduction, using the mole concept involved in stoichiometry with appropriate treatment by dimensional analysis or unit cancellation, was also visually displayed along with proper narration. After viewing the video portion, data collection and treatment from an actual experiment should be more readily and accurately handled. The computer segment for the laboratory experiment was made to offer three options: (1) general drill on stoichiometry, using potassium chlorate (as opposed to the trite A_2B_3,etc., type generally put in some computer programs); (2) specific drill on realistic data that could be obtained from the actual laboratory experiment, with the limits of weights given in the exercise; and (3) simulated, random, but again, realistic data generated by the computer. During the drill, if problems did arise, the student was given some instructional hints as well as directions to go back to the video segment (usually to a certain counter number on the video tape machine to avoid viewing the entire video tape). In this way, the student would be encouraged to try again, with almost guaranteed results for success, if directions were followed. Any or all of the options given could, of course, be selected for use.

The CAVE program for the laboratory experiment Spectrophotometry was a bit different. It involved a separate computer program in addition to the computer program designed to augment the video segment. The first computer program was a brief drill on basic principles of visible spectrophotometry, with emphasis on Beer's Law and its usage in both solution identification and determination of concentrations. The video segment probed the insides of the visual spectrophotometer -- a black-box mystery to most students -- with emphasis on how the device works and how to use it properly. Here too, the video portion offered close-up detail for operational efficiency, which in some cases could not be performed adequately for a large group, nor as effectively. The computer program again simulated the laboratory procedure with appropriate drill on the data, which was also realistic and randomly generated.

The final results based on student usage will be available by May from the Spectrophotometry programs (since the laboratory experiment is scheduled for the second semester). Some encouraging results have already been obtained from the use of the Stoichiometry of Potassium Chlorate programs. Ten students were selected for the use of this latter package. All commented favorably on its usage. Three students from our classes did miss the experiment and were directed to use the program package, handing in the computer printout of the simulated data along with the laboratory exercise pages with the calculations made using these data. The results are summarized below:

Average Mark Using CAVE (13 Students)	Average Mark Doing Actual Experiment (29 Students)
94.5	81.0

In addition, a problem was used on an hour exam which was a repeat of one part of data collection and treatment in the experiment. The percent of students getting the problem right with and without the use of the program were 76% and 60%, respectively.

These results are encouraging, although admittedly with such a small sampling, they may be misleading. Further evaluation in this area during the early part of 1978 will allow me to make more decisive comments by the time of the conference. We hope that we can encourage more usage of the program in next year's classes to get a more meaningful result.

We would like to continue the CAVE project in the laboratory line, with emphasis on those experiments which are both strong supplementers of classroom work and which have proven somewhat troublesome in the past. The video and computer programs have, we believe, shown great promise in this marriage of the two media. (Perhaps it is only some seasoning added to our main dish, in that some may desire to omit it, but it certainly deserves a place on the shelf of every gourmet teacher.)

Our special thanks to Professor J. C. Thompson and Mr. Al Weiss of the Educational Resources Center at the United States Naval Academy; as well as to Professor Richard A. Pollak and Mr. James Schwab of the Academic Computing Center; and to their respective staffs for technical and moral support. This work was supported by a Naval Academy Research Grant from the office of the Academic Dean, Dr. Bruce M. Davidson, U. S. Naval Academy.

COMPUTER AUGMENTED VIDEO EDUCATION (CAVE) IN TROUBLESOME AREAS OF CHEMISTRY

by

John V. Prestia
Chemistry Department
U.S. Naval Academy
Annapolis, Maryland 21402

During the past several years, the chemistry faculty of the U. S. Naval Academy has prepared videotaped programs covering a wide variety of topics normally included in a first-year college general chemistry course. These programs have been used in several ways, but their use as tutorial material by the students appears to be the most fruitful and interesting application. In chemistry, as in any discipline in which quantitative calculations are involved, it is necessary that a student spend some time in practicing quantitative manipulations in order to acquire a better understanding of the material. It is not unusual for a student to "understand" when an instructor works a sample problem, only to find that he is stymied when he attempts to work one himself. The computer can be a powerful educational tool when it is used to provide students with this needed drill and practice. The CAVE project now makes it possible to augment the videotaped tutorials so that the student may participate by actual problem solving and be offered help as he progresses through a particular sequence. Thus we are taking advantage of both the interactive capabilities of the computer and the audio-visual nature of television to provide one powerful instructional resource.

Beginning general chemistry students quite commonly have difficulty in areas where new concepts and new terms are involved in basically simple calculations. Molecular Weight, Mole, Ideal as opposed to Real gases, and Hydrolysis are only examples from a long list that any instructor of chemistry can produce in a matter of minutes. The object of the CAVE project as it applies to these traditionally troublesome areas is to provide the student with an audio-visual presentation coupled with an opportunity for direct involvement by working sample problems. By allowing him to proceed at his own pace, to review as often as he feel necessary, and to utilize help sequences, CAVE permits the deficient but dedicated student to obtain remedial instruction when and where he needs it. During the first phases of this work, we have identified several topics that often give the student trouble and for which we have developed CAVE programs. Initially, we analyzed these topics in terms of goals to be attained and design of the package. Some of these outlines are shown in the appendix.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

These outlines can be summed up as follows:

1. Audio-Visual presentation of a concept.
2. Sample problem.
3. Student practice.
4. Help sequence.

Normally only two or three concepts are contained in a single program so that the television portion would be no longer than 15 or 20 minutes. This allows the student ample time to review and practice and still stay within a classroom hour, to which he may be limited by his schedule.

The hardware for the use of these programs would consist of a videocassette playback machine and computer keyboard, both hooked to a single television monitor. After selection of the tape to be used, a student follows the printed instructions to start the program. He is instructed to switch to the computer terminal, by means of a single selector switch, and to sign on using his identification number and password. He then returns to the video tape for synchronization by depressing a counter button at the designated time. When the student is ready for the drill and practice portion, he depresses the transfer switch which holds the video tape at that point and puts the system into the computer mode. The following presentation then appears on the CRT:

◆ ◆ ◆ ◆ ◆

```
     THIS PROGRAM OFFERS THE STUDENT
SOME SPECIAL FUNCTION OPERATIONS THAT
MAY BE TURNED ON OR OFF BY THE QUESTION
AUTHOR.  THE STUDENT MAY USE ANY OF THE
FUNCTIONS THAT ARE TURNED ON ANYTIME THE
PROGRAM IS WAITING FOR AN ANSWER.

TO USE A FUNCTION - TYPE IN THE CODE
FOR THE DESIRED FUNCTION INSTEAD OF
THE ANSWER.

THE LEGAL FUNCTIONS ARE:

CODE  FUNCTION DESCRIPTION     STATUS
$ST   TO STOP THE PROGRAM
      RUN BEFORE COMPLETION      ON
$CA   TO PUT THE TERMINAL IN
      'CALCULATOR MODE' TO
      PERFORM CALCULATIONS       ON
$QU   TO REPRINT THE QUESTION    ON
$SE   TO SELECT BY FRAME # THE
      NEXT QUESTION              ON
$SK   TO SKIP TO THE NEXT QUEST-
      ION IN SEQUENCE            ON
$BE   TO SKIP TO THE BEGINNING
      OF THE EXERCISE            ON
$IN   TO GET THESE INSTRUCTIONS  ON

THIS PROGRAM WILL PERMIT YOU TO
PRACTICE WT-VOL AND VOL-VOL PROBLEMS.
YOU MAY WISH TO COPY DOWN SOME NUMBERS.
```

```
R, THE UNIVERSAL GAS CONSTANT, EQUALS
    0.0821 LITER-ATM/DEG KELVIN-MOLE.
GMV, THE GRAM MOLECULAR VOLUME, IS
    22.4 LITERS AT STP.
**REMEMBER THAT YOU CAN UE THE 22.4
ONLY WHEN YOU ARE WORKIG WITH GASES
AT ZERO DEG C AND 1 ATM PRESSURE****

CONTINUE WHEN READY BY PUSHING RETURN.

YOU MAY NOW CHOOSE BETWEEN TWO TYPES OF
PROBLEMS.
TYPE A- WT-VOL AND VOL-WT PROBLEMS.

TYPE B - VOL-VOL PROBLEMS.
MAKE YOUR CHOICE BY TYPING A OR B.
ANSWER? a

IN THIS BRANCH OF THE PROGRAM, YOU WILL
BE GIVEN WT-VOL PROBLEMS THEN VOL-WT
PROBLEMS (TWO OF EACH). YOU MAY SKIP
A PROBLEM BY TYPING $SK, BUT REMEMBER
THAT YOU ARE HERE BECAUSE YOU NEED HELP.
```

◆ ◆ ◆ ◆ ◆

The first problem is then presented, having been selected in a random fashion by the computer. Thus, even though a student chooses to use the same program more than once, he will be given different practice problems. The first example sequence that follows shows the use of the calculator mode, a correct response, and several incorrect responses. Note that the program identifies the more common types of errors, e.g., centigrade temperature instead of kelvin. If the student is still not able to answer the question after a predetermined number of trials, he is given the correct answer and the opportunity to follow through with a help sequence. He may at any time return to any portion of either the video tape or the computer program.

◆ ◆ ◆ ◆ ◆

```
QUESTION # 1   FRAME 6

FIRST WE NEED A BALANCED EQUATION.
N2 + 3H2 -> 2NH3
USING PV = (MOLES)RT
R = .0821 LITER-ATM/DEG KELVIN-MOLE

FIND THE VOLUME OF H2
REQUIRED TO REACT WITH
174 GRAMS OF N2
ASSUME COMPLETE REACTION..
GAS MEASURED AT 47.5 DEGREES CENT.
AND 0.95 ATOMOSPHERES PRESSURE.
ANSWER?  $ca

ENTERING 'CAL'MODE!
HIT THE CARRIAGE RETURN AFTER '->'
TO GET OUT OF CAL MODE AND RETURN
TO ANSWER THE QUESTION.
->174/28*3

 18.6429
->18.6*.0821*(273+47.5)/.95

 515.182
->
```

```
ANSWER?  515

SPLENDID!

QUESTION #2    FRAME 6

FIRST WE NEED A BALANCED EQUATION.
2KCL03 -> 2KCL + 302
USING PV = (MOLES)RT
R = .0821 LITER-ATM/DEG KELVIN-MOLE

FIND THE VOLUME OF 02
PRODUCED WHEN
21.7 GRAMS OF KCL03
ARE COMPLETELY DECOMPOSED..
GAS MEASURED AT 34.4 DEGREES CENT.
AND 0.929 ATOMOSPHERES PRESSURE.
ANSWER?

ANSWER MUST BE NUMERIC!  ENTER AGAIN...
? $ca

ENTERING 'CAL' MODE!
->21.7/122*3/2

 0.266803
->.267*.0821*34.4/.929

 0.811703
->

ANSWER?   .812

YOU PROBABLY USED CENT TEMP VICE KELVIN
ANSWER?  234

SORRY, ITS LOOKS LIKE YOU TRIED TO
USE THE MOL WT OF THE GAS -NOT NEEDED.
LAST CHANCE ON THIS QUESTION!
? 56

INCORRECT # OF SIGNIFICANT DIGITS!

THE CORRECT ANSWER IS:
21.7/123*3/2*.0821*(34.4+273)/0.929 = 7.19 E0

YOU SEEM TO BE HAVING SOME TROUBLE.
YOU MAY CHOOSE BETWEEN A HELP SEQUENCE
HERE ON THE COMPUTER OR YOU MAY RETURN
TO THE TV TAPE AT COUNTER NO. XXXX TO
REVIEW THE MATERIAL AGAIN.
TYPE COMPUTER OR TV.
ANSWER?  computer

YOU ARE OBVIOUSLY HAVING SOME TROUBLE.
FOLLOW THE STEPWISE DEVELOPMENT HERE.

2KCL03 -> 2KCL + 302
27.9 GRAMS OF KCL03
REACT COMPLETELY AS SHOWN.
CALCULATE THE VOL OF 02 PRODUCED..
FIRST FIND THE NUMBER OF MOLES OF KCL03
DIVIDE BY THE MOLECULAR WEIGHT (123).
27.9/123 = 0.227
NEXT WE USE THE COEFFICIENTS OF THE
BALANCED EQUATION TO GO FROM MOLES OF
KCL03 to MOLES OF 02.
0.227*3/2 =0.34
HIT THE RETURN TO CONTINUE
```

```
NOW THAT WE HAVE MOLES OF THE GAS, WE
CAN USE PV = (MOLES)RT TO CALCULATE
THE VOLUME (V) AT ANY TEMP AND PRESS,
OR GO DIRECTLY FROM MOLES OF THE
GAS TO VOLUME AT STP BY
MULTIPLYING MOLES BY 22.4 (GMV).
REMEMBER TO USE T IN DEGREES KELVIN
AND THAT THE UNITS FOR R MUST BE IN
AGREEMENT WITH THE UNITS FOR P AND V.
```

◆ ◆ ◆ ◆ ◆

Initial student comments indicate that the CAVE concept will be used extensively as soon as sufficient software is available. At the time this is being written (late December '77, only a limited number of programs are operational, and the student is not able to get help in all of the areas that he needs. Since the initial work is now complete, it is anticipated that a single program will require about two man-weeks of effort to prepare. We feel that it would be more efficient to have more than one person on the project so that individuals may devote their time to specific aspects of the development. We hope to have about fifteen instructional packages available for the 1978-79 academic year.

APPENDIX

Instruction Package #SC9501: Quantitative Aspects of Chemical Reactions

GOAL	INSTRUCTIONAL OBJECTIVES	INSTRUCTIONAL DESIGN
To have the student understand the mole concept and to teach him to work weight-weight problems.	To know:	Videotape
	a) Definition of mole. b) Avogardo number. c) Meaning of chem subscripts. d) Meaning of chem equations.	1. Definitions or explanations a) Mole b) Avogadro number c) Chemical Subscripts d) Chemical equations
	To calculate:	2. Sample problems
	a) Gram-mole problem. b) Gram-number of atoms problem. c) Weight-weight problem. d) Weight-mole problem.	a) Grams to moles b) Moles to grams c) Weight-weight problem
		Computer
		Student will work: a) Grams to moles b) Moles to grams c) Grams to number of atoms d) Number of atoms to grams e) Weight-Weight problem

Instruction Package #SC9505: Gases in Stoiciometric Chemical Calculations

GOAL	INSTRUCTIONAL OBJECTIVES	INSTRUCTIONAL DESIGN
Expand Package I to include Weight-Volume problem	To know: a) How and when to use GMV. b) Volume relationship for gases given by balanced equation. To calculate: a) Weight-volume problem. b) Volume-volume problem.	Videotape 1. Definitions or explanations a) Conditions for use of GMV b) Volume relationship for gases from equation c) Avogadro Principle 2. Sample problems a) Weight-volume problem b) Volume-weight problem Computer Student will work: a) Weight-volume b) Volume-weight

Instruction Package #SC9506: Equation of State for Gases

GOAL	INSTRUCTIONAL OBJECTIVES	INSTRUCTIONAL DESIGN
To have the student understand and use the Ideal Gas Equation and to understand the origin and use of the Gram Molecular Volume.	To know: a) Ideal Gas Equation. b) Laws from which Ideal Gas Equation is derived. c) Origin and use of GMV. d) Limitations of $PV=nRT$ To calculate: a) Any one of P, V, n, or T when other three are given. b) Volume of gases from moles. c) Moles from volume of gases.	Videotape 1. Definitions or explanations: a) Ideal vs Real Gas. b) $PV = nRT$. c) Determination of R. d) Determination of GMV. 2. Sample problems: a) None. Computer Student will work: a) P,V,N and T problems. b) Mole-Volume problems.

Appendix

Instruction Package #SC9507: Weak Acid-Weak Base Calculations

GOAL	INSTRUCTIONAL OBJECTIVES	INSTRUCTIONAL DESIGN
To review the principles of equilibrium as they apply to weak acids and weak bases.	To know: a) Weak Acids and Bases from Strong Acids and Bases. b) Form of equilibrium condition, K_a or K_b. To calculate: a) K_a from (H^+). b) (H^+) from K_a. c) Same for bases.	Videotape 1. Definitions or explanations: a) Weak Acids and Weak Bases. b) Equilibrium conditions. c) Symbols and simplifications. 2. Sample problems: a) K_a from pH. b) K_b from pH. c) (H^+) from K_a. Computer Student will work: a) K_a from pH b) K_b from pH c) (H^+) from K_a for acids. d) (OH^-) from K_b for bases.

Instruction Package #SC9512: Hydrolysis of salts

GOAL	INSTRUCTIONAL OBJECTIVES	INSTRUCTIONAL DESIGN
To discuss the acidic and basic properties of ions.	To know: a) Acid or basic character of salts. b) Relationship between K_a for weak acid and K_b for corresponding conjugate base. To calculate: a) K_a from K_b for conjugate acid-base pair. b) pH from K_a or K_b for ions. c) K_a or K_b for ions from pH	Videotape 1. Definitions or explanations a) Hydrolysis b) K_a-K_b relationship for acid base pair c) Relative strengths of acid and base in a pair. 2. Sample problem a) K_b knowing K_a of conjugate acid Computer Student will: a) Determine acid-base properties of salts. b) Calculate pH from K_a or K_b of acid or base (ions). Hydrolysis problems.

Appendix

Instruction Package #SC9513: BUFFERS

GOAL	INSTRUCTIONAL OBJECTIVES	INSTRUCTIONAL DESIGN
To describe the nature of buffer systems and the associated calculations	To know:	Videotape
	a) Requirements for a buffer.	1. Definitions or explanations
	b) How to prepare a buffer.	a) Buffer b) Buffer operation
	c) Application of LeChatelier's	2. Sample problem
	To calculate:	a) pH of a buffer
	a) pH of a buffer solution.	Computer
		Student will:
		a) Determine pH of a buffer. b) Determine amount of acid and salt present - as prepared by three different methods.

Appendix

THE EFFECTS OF CAI ON ACHIEVEMENT AND ATTITUDE IN THE FRESHMAN SURVEY MATHEMATICS CURRICULUM

Dr. Leo Edwards, Jr.

Fayetteville State University

The Mathematics Department at Fayetteville State University is committed to provide an opportunity for every student to develop techniques of intellectual inquiry. To facilitate our mission, research has become the tool of guidance and pacing, and computer assisted instruction seems to be indicative of a viable course.

Statement of the Problem

Members of the department are constantly asking, "What can we do when students at the college level cannot perform basic operations? To whom do I gear my instruction when a given class has a set of students totally unprepared? What about the time it takes to deal with basic operations?" Thus, the purpose of this investigation was to seek answers to these perennial questions; a review of the literature in mathematics education provided some solace.

Review of the Research Literature

There are several studies that have delt with the relationship between attitude and achievement in mathematics at all levels of schooling: the elementary (Moore, 1972), secondary (Burbank, 1968; Callahan, 1971; Edwards, 1976; Wilson, 1973) and post-graduate (Webb, 1971). Occasionally, a study such as Webb (1971), finds attitude to be the most important predictor of success or lack of success in mathematical achievement.

Using Webb's (1971) findings as a point of departure for the instructional investigation, a review was made on the effects of calculators and other physical materials toward the improvement of attitude toward mathematics (Cech, 1972; Gaslin, 1972). The results of a number of studies show similar improvements in attitude with the "calculator" groups and the "no calculator" groups when simple machines are employed.

Crawford (1970) reported more favorable attitudes toward mathematics as a result of computer assisted instruction (CAI) with seventh graders. While some research is indicative of highly significant effects of machines, techniques and/ or materials on attitude (Burgess, 1969), the results of other reports are disappointing (Brown, 1972; Simpson, 1973).

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

To enhance the possibility of engendering a positive attitude toward the study of mathematics, a social learning theory was envisioned as a fascimile to the design described by Rotter (1966). He suggested that in our culture if a reinforcement is perceived by the subject as following some action of his own but not being entirely contingent upon his action it is typically perceived as the result of luck, chance, fate, as under the control of others who are powerful, or an unpredictable because of the great complexity of the forces surrounding him. When the event is interpreted in this way by an individual, it is labeled a belief in external control. If the person percieves that the event is contingent on his own actions, it is labeled a belief in internal control. Analogous to the views of Rotter (1966) are those of Vasquez (1974) who considers the understanding of cause-effect relationships as essential. The internalized attribution of success or failure on the part of the student (positive or negative attitude), Vasquez (1974) further suggests four criteria by which to evaluate activities to accomplish understanding of the cause-effect relationship:

1. See that the cause has sufficient power to produce the effect.
2. Students should comprehend that without the cause, the action or result would not have occurred.
3. Other equally likely causes must not be present if we are to identify a single cause.
4. The cause must precede the event in time.

Method and Organization

Recognizing the criteria suggested by Vasquez (1974) as guides for curriculum development in freshman mathematics, the Mathematics Department embarked upon a program incorporating computer literacy and computer assisted instruction (CAI). The computer literacy program had all students in MAT 112 (the terminating liberal arts mathematics course and sequel of MAT 111) using the HP 2000 time sharing system via the BASIC Programming Language. CAI students (MAT 111 - a randomly selected section) also used BASIC, but were given identification number for a canned program that initiates with basic or fundamental operations. These students attended classes for a three (3) credit course two (2) periods a week, and spent at least thirty (30) minutes a week at the terminals while studying the module on basic operations (MAT 111 is composed of five modules). Although the time indicated for completion of the module was two weeks, the students were given the option of completing the computer assignments by final examination week. They could put in their terminal time any period of the day from 8:00 a.m. to 7:30 p.m. Monday through Thursday and 8:00 a.m. to 5:00 p.m. on Friday. The programs are set up in blocks with a set of lessons in each block keyed to a pretest and post-test. The pre-test score determines which lesson-set the student will begin; the set will repeat a given question two times for the student to answer. If on the second chance the student misses the question, the computer will repeat the question a third time and give the answer. It should be noted that a block is analogous to a chapter in a textbook whereby tests are given on various segments of work within a block, followed by a block test. If the student is less than 80% accurate on a given test, review material is generated (sometimes before/after moving to the subsequent block).

Typically, the students (CAI) were registered into the computer by their instructor for work periods in intervals of fifteen minutes. This was done to facilitate campus use of only two terminals. As was mentioned earlier, the student gained access to his program via his ID number -- first name. If the student, 2345, JOHN, stopped at point "C" in a given block during his last work period, the computer automatically starts at a point "C" level of difficulty in the same block on his next work effort. Each student in CAI had to cover the (10) blocks before it could be said he had satisfactorily completed fundamental or basic operations. Ten (10) blocks of work adequately covered fundamentals of arithmetic mechanics.

Hypotheses

The following null hypotheses were tested:

1. There is no difference in the attitudes among students who received the CAI instructional method and students who received the traditional method as measured by the Mathematics Attitude Scale.
2. There is no difference in the performance of fundamental operations among students who received the CAI instructional method and students who received the traditional method as measured by the Algebra Readiness Test.

Description of Measures Employed

Two instruments were used in the collection of the data for this investigation. One of the instruments, the Mathematics Attitude Scale (Aiken, 1972) was employed to provide a general description of "enjoyment of mathematics"--which encompasses not only a liking for mathematical problems, but for mathematical terms, symbols, and routine computations. The test consists of 20 questions of which the correlation coefficient of reliability of .95 and the predictive validity is listed as .40. This instrument was used after the treatment to assess attitudes. The time required to administer the Mathematics Attitude Scale is approximately 10 minutes.

The other instruments, Algebra Readiness Test, is a battery consisting of five sub-tests and six scores (fundamental operations, fractions, decimals, problem solving, general numbers and total). For investigative purposes, only the first three sub-tests were administered. Hence, a summation of the raw scores obtained for the three sections was made for the investigative attempts of "fundamental operations." The reliability coefficient of this test is given as .96 while the validity coefficient is given as .70. The time required for the administration of the Algebra Readiness Test is 30 minutes (Lueck, 1947).

Statistical Design and Analysis

A post-test only control group design was used to facilitate the analysis of data (Borg, 1971). A description of the design is displayed in Table I.

TABLE 1

Post-test - only Control Group Design

R	X	O
R		O

Key: R = Random Assignment
X = Experimental
O = Observation

Student's-t distribution techniques were employed as a test for significance at the .05 level, using unequal variance. The results are summarized in Table 2.

TABLE 2

Student's-t Distribution

Sample	Sample Data Size	Mean	Standard Deviation
Experimental (3 x 23)	69	9.72	6.36
Control (3 x 29)	87	7.89	5.42

($p < .05$)
$t = 1.91$ at 135.44 df

Results and Implications

The null hypothesis was rejected with respect to fundamental operations in favor of the experimental group. The critical value is 1.65, which is significant, although the level is not extremely impressive. More importantly, the significance is indicative of a need for more CAI and research.

The data contained in Table 3 are representative of the mean comparisons on attitudes.

TABLE 3

Student's-t Distribution
Attitude

Sample	Sample Data Size	Mean	Standard Deviation
Experimental	23	5.3478	18.37
Control	29	60.85	17.49

($p < .05$)
$t = 1.67$ at 47.88 df

The critical region for this component (attitude) of the investigation is to the left of 1.645, which means that CAI enhanced positive attitudes in the freshman mathematics students.

It is also felt that a more positive attitude could have been engendered had the department been able to afford more than two (2) terminals.

Further Implications and Summary

The CAI system as previously discussed also has administrative capabilities for daily progress reports. Such reports can free an instructor to work with students who would not ordinarily get attention, and who desperately need attention.

It is strongly felt by the researcher that there is cause-effect relationship with CAI (immediate feedback).

With CAI, the student instantly recognizes the relationship of the output to his input. Moreover, the interaction between the student and the terminal is direct. Thus, the student must take full responsibility or credit for the response. The student is cut off from external control, and has all actions internally controlled.

Recommendations

The researcher strongly recommends:

1. CAI for marginal students at all levels of schooling.
2. The purchase of time-sharing and terminals for school districts and independent schools at all levels.
3. Affective strategies in the dissemination process.
4. Additional research to be conducted on CAI at all academic levels.

REFERENCES

Aiken, Louis R. 1972 "Biodata Correlates of Attitudes Toward Mathematics in Three Age and Two Sex Groups". School Science and Mathematics. 72: 386-395.

Borg, W. 1973. Educational Research: An Introduction. David McKay Co., Inc., New York. 388-390.

Brown, J. D. 1972. "An Evaluation of the Spitz Student Response System in Teaching a Course in Logical and Mathematical Concepts." Journal of Experimental Education. 40: 12-20.

Burbank, D. K. 1968. "Relationships Between Parental Attitude Toward Mathematics and Student Attitude Toward Mathematics, and Between Student Attitude in Mathematics and Student Achievement in Mathematics". Dissertation Abstracts, 1970, 30: 3359-A-3360-A.

Burgess, E. E. 1969. "A Study of the Effectiveness of the Planned Usage of Mathematical Games on the Learning of Skills and Concepts and on the Attitude Toward Mathematics and the Learning of Mathematics of Low Achieving Secondary Students." Dissertation Abstracts, 1970, 30: 533a-534A.

Callahan, W. J. 1971. "Adolescent Attitude Toward Mathematics." The Mathematics Teacher. 64 (12): 751-755.

Cech, J. P. 1972. "Effect of the Use of Desk Calculators on Attitude and Achievement with Low Achieving Ninth-Graders." Mathematics Teacher. 63 (3): 183-186.

Crawford, A. N. 1970. "A Pilot Study of Computer-Assisted Drill and Practice in Seventh-Grade Remedial Mathematics." California Journal of Educational Research. 21: 170-181.

Edward, R. R. 1972. "Predication of Success in Remedial Mathematics Courses in the Public Community Junior Colleges." Journal of Educational Research. 66: 175-160.

Edwards, L. 1976. "The Effects of a Problem Solving Model as an Alternative in the General Mathematics Curriculum". Unpublished Doctoral Dissertation. Utah State University.

Fennerman, G. C. 1973. "The Validity of Previous Experience, Aptitude, and Attitude Toward Mathematics as Predictors of Achievement in Freshman Mathematics at Wartburg College," Dissertation Abstracts, 1974, 34: 71-A-7101-A.

Gaslin, W. L. 1972. "A Comparison of Achievement and Attitudes of Students Using Conventional or Calculator Based Algorithms for Operations on Positive Rational Numbers in Ninth-Grade General Mathematics." Dissertation Abstracts, 1972, 33: 3317-A.

Moore, B. D. 1971. "The Relationship of Fifth-Grade Students' Self Concepts and Attitudes Toward Mathematics to Academic Achievement in Arithmetical Computation, Concepts, and Application." Dissertation Abstracts, 1972, 32: 4426-A.

Rotter, J. B. 1966. Generalized Expectancies for Internal Versus External Control of Reinforcement. PSYCHOLOGICAL MONOGRAPHS, 1966, 80 (1, While No. 609.)

Simpson, C. J. 1973. "The Effect of Laboratory Instruction on the Achievement and Attitudes of Slow-Learners in Mathematics." Dissertation Abstracts, 1974, 34: 6949A-6960A.

Spickerman, N. R. 1965. "A Study of the Relationship Between Attitudes Toward Mathematics and Some Selected Pupil Characteristics in a Kentucky High School." Dissertation Abstracts, 1970, 30: 2733-A.

Vasquez, James, 1976. "Learning in Relation to Desegregated Classrooms", STRIDE, June, 1975, General Assistance Center, San Francisco, Ca.

Vasquez, James A. 1976. "Locus of Control Learning and Implications for Educators:, An Unpublished Manuscript presented at the General Assistance Center, Center, 1974, San Francisco, Ca 94103.

Webb, RJ. 1971. "A Study of the Effects of Anxiety and Attitudes Upon Achievement in Doctoral Education Statistics Course." Dissertation Abstracts, 1973, 32: 4997-A-4998-A.

Wilson, J. M. 1973. "Post Mathematical Attitude Among Prospective Elementary Teachers as Predicated by General Mathematics Skills. Modern Mathematics Achievement and Prior Mathematical Attitudes." Dissertation Abstracts, 1973, 34: 3453-A

A GENERAL PROBLEM SOLVER FOR MATHEMATICS COURSES

Roger C. Barr[1]
Thomas M. Gallie, Jr.[2]
Margaret J. Hodel
Richard E. Hodel
Francis J. Murray
David A. Smith[3]

Department of Mathematics
Duke University
Durham, N. C. 27706
919-684-2321

Introduction

We reported in another forum [4] over two years ago on a new approach to the use of computers in mathematics courses. At that time we had just begun a new freshman calculus course utilizing this approach. We can now report on three years of successful implementation in that course, plus some additional experience with other courses.

Our objectives in this project are to provide high quality computer service to our students; to do so at low cost, without additional investment in hardware; and to avoid investing time and effort in teaching programming or competing with the Computer Science Department, which properly has the responsibility for teaching programming. On the other hand, we did not want to provide canned programs for each numerical method used, since adequate teaching of algorithms requires a substantial involvement by the students with the algorithms themselves.

These objectives are being met through the use of a very general and flexible problem solver, the MATHPROGRAM. The program is merely a sequence tabulator, capable of tabulating any specified number of terms of up to seven interrelated sequences, each defined either recursively or explicitly. (Details of the program are provided in the next section.) In a sense, this accounts for all computational problems in mathematics. (However, we have made no

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

attempt to accomodate vector and matrix sequences, except in a very limited sense described below.)

Duke obtains computer services from Triangle Universities Computation Center (TUCC), which provides a wide spectrum of batch and interactive capabilities on two IBM 370/165's and an HP 2000. The batch capabilities include the student-oriented autobatch systems WATFIV and PL/C, and the interactive systems include APL, TSO, CPS, and BASIC. The Duke Computation Center has one high speed and four medium speed batch terminals to TUCC, and there are approximately 10 public, low speed, interactive terminals on campus, all in heavy demand.

For four years prior to this project, one of us (D.A.S.) was offering a computer-oriented supplement to the calculus course, limited to 15 students, drawn from more than 900 in our standard course. These students learned to write their own short programs in CPS or BASIC. (See [2] for a description of the course.) Notes prepared for these students eventually grew into a textbook [5], which we still use in the very different MATHPROGRAM environment. A shift away from interactive computing was dictated by a decision made in the fall of 1974 to expand our computer-calculus offering to a wider audience, but without spending more money on computer es or hardware. When we studied the problems of implementing ecision, MATHPROGRAM was the result.

While ease of access and use by students was crucial for the ss of our new course, it was equally important that the course achable by members of the department with no prior experience teaching computer-based mathematics. It could no longer be a "one man show", especially since the instructor in the previous course was to be on leave during the first year of the new course. Two of the present authors (M.J.H., R.E.H.) were in the "no prior experience" category and have subsequently taught the course with no ill effects, indeed with considerable satisfaction.

During the spring term of 1975, weekly "training sessions" in the use of the program and related mathematical topics were held for prospective teachers of the new course. This also provided a real test of the simplicity and generality of the program before confronting students with it. Since the fall of 1975, the course has been offered as an option to entering freshmen, who preregister before arriving on campus. The course parallels our standard, 3-hour-per-week course, following the same syllabus and using the same uniform final exams, but there is an extra class period per week (lab session) in which the computer related topics are discussed.

Over the past three years we have averaged 48 students in the fall term, 34 in the spring term in this course. These students have produced better than average results on our block final exams, but that is just as likely to result from the self-selection process as from anything extra they may have learned by using a computer. We have used two very different calculus books [9], [8] during these three years. The change of texts required some rearrangement of the problems, but we see no barrier to the use of this material to supplement any calculus book.

MATHPROGRAM was also used during the sabbatical leave mentioned above with a group of 10 students at Case Western Reserve University in a January term course (1976). Each of these students completed 10 computer problems and a substantial project in about three weeks.

The course had the blessing and encouragement of the CWRU mathematics department, but it was rather subversive in that environment. All of their freshman calculus students were required to learn Algol (hampered by the primitive IBM 026 character set and by many calculus instructors who didn't really want to learn or teach programming), and to complete a computer assignment every two weeks. It was quite an eye opener for the students to discover that a single program would solve all the problems they had struggled with during the first term or would see in the second.

We have since found that MATHPROGRAM has many other uses beyond freshman calculus. It has been used with a large section of our third semester calculus course, with a small group in a complex variable course, and for various small tasks. These experiences are also reported below.

The program as we use it is written in the PL/C dialect of PL/I and run in autobatch mode. The cost per run is close to the minimum cost for compiling and running anything on our system (30 to 40 cents), about half the cost of comparable runs in WATFIV. For reasons that will be clear below, students can often solve a problem with a single run and seldom need more than two or three runs. The PL/C system has excellent error diagnostics, and there are only a handful of places at which a student can make an error.

A version of MATHPROGRAM in ANSI Standard FORTRAN has been prepared for us by the staff of CONDUIT. Sometime in 1978, a package consisting of this transportable program, plus student and instructor manuals, will be submitted to the CONDUIT review and certification procedure.

The Program

Here is the algorithm for MATHPROGRAM:

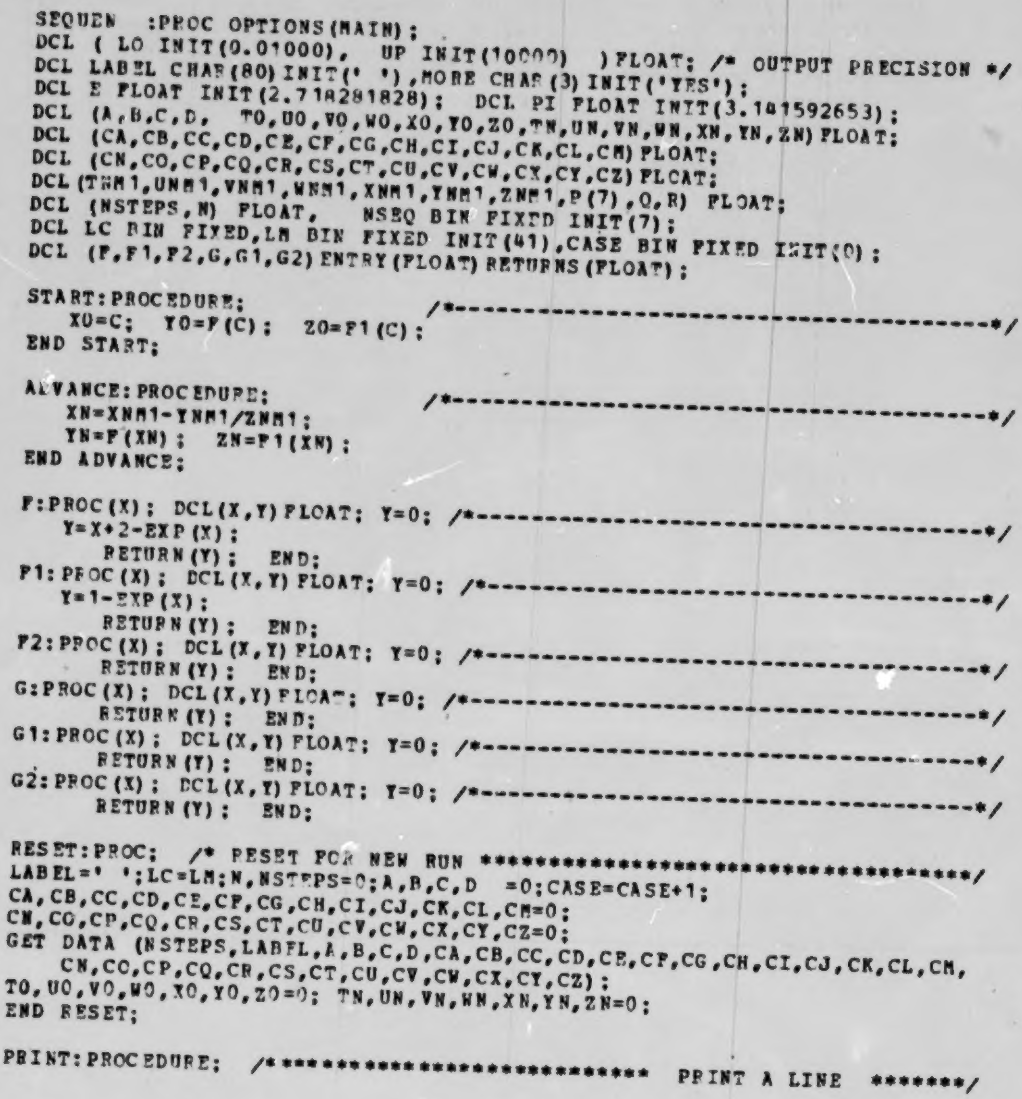
```
(1)  Initialize:  Set all variables to zero and read input data.
(2)  Call START (initialize sequences).
(3)  Set TNM1 = T0, UNM1 = U0,...,ZNM1 = Z0, and print line zero.
(4)  For  N = 1,2,3,..., NSTEPS, do:
     (a)  Call ADVANCE (to advance sequences).
     (b)  Set TNM1 = TN,...,ZNM1 = ZN, and print line N.
(5)  Return to step (1).
```

Our implementation of this simple algorithm in PL/C is shown in Figure 1, with the deck set up to find a root of $f(x) = 0$ by Newton's method. The output is shown in Figure 2. These figures illustrate the important features of the program.

The unnumbered cards in the deck are the user-supplied cards in the START, ADVANCE, and function procedures. Three of the seven sequences are used: XN for the successive approximations x_n of Newton's method, YN for $f(x_n)$, and ZN for $f'(x_n)$. The START statement X0 = C identifies the initial guess as an input item, which permits location of all the roots on a single run by stacking data cards with different starting points. The heart of the algorithm is the ADVANCE statement

XN = XNM1 - YNM1/ZNM1;

Note following the ADVANCE procedure that the user has the choice



```
SEQUEN   :PROC OPTIONS(MAIN);
DCL ( LO INIT(0.01000),  UP INIT(10000)  )FLOAT; /* OUTPUT PRECISION */
DCL LABEL CHAR(80)INIT(' '),MORE CHAR(3) INIT('YES');
DCL E FLOAT INIT(2.718281828);  DCL PI FLOAT INIT(3.141592653);
DCL (A,B,C,D,  T0,U0,V0,W0,X0,Y0,Z0,TN,UN,VN,WN,XN,YN,ZN)FLOAT;
DCL (CA,CB,CC,CD,CE,CF,CG,CH,CI,CJ,CK,CL,CM)FLOAT;
DCL (CN,CO,CP,CQ,CR,CS,CT,CU,CV,CW,CX,CY,CZ)FLOAT;
DCL(TNM1,UNM1,VNM1,WNM1,XNM1,YNM1,ZNM1,P(7),Q,R) FLOAT;
DCL (NSTEPS,N) FLOAT,   NSEQ BIN FIXED INIT(7);
DCL LC BIN FIXED,LM BIN FIXED INIT(41),CASE BIN FIXED INIT(0);
DCL (F,F1,F2,G,G1,G2)ENTRY(FLOAT)RETURNS(FLOAT);

START:PROCEDURE;                   /*-------------------------------------*/
   X0=C;  Y0=F(C);  Z0=F1(C);
END START;

ADVANCE:PROCEDURE;                 /*-------------------------------------*/
   XN=XNM1-YNM1/ZNM1;
   YN=F(XN);  ZN=F1(XN);
END ADVANCE;

F:PROC(X); DCL(X,Y)FLOAT; Y=0; /*-----------------------------------------*/
   Y=X+2-EXP(X);
      RETURN(Y);  END;
F1:PROC(X); DCL(X,Y)FLOAT; Y=0; /*----------------------------------------*/
   Y=1-EXP(X);
      RETURN(Y);  END;
F2:PROC(X); DCL(X,Y)FLOAT; Y=0; /*----------------------------------------*/
      RETURN(Y);  END;
G:PROC(X); DCL(X,Y)FLOAT; Y=0; /*-----------------------------------------*/
      RETURN(Y);  END;
G1:PROC(X); DCL(X,Y)FLOAT; Y=0; /*----------------------------------------*/
      RETURN(Y);  END;
G2:PROC(X); DCL(X,Y)FLOAT; Y=0; /*----------------------------------------*/
      RETURN(Y);  END;

RESET:PROC;  /* RESET FOR NEW RUN ****************************************/
LABEL=' ';LC=LM;N,NSTEPS=0;A,B,C,D  =0;CASE=CASE+1;
CA,CB,CC,CD,CE,CF,CG,CH,CI,CJ,CK,CL,CM=0;
CN,CO,CP,CQ,CR,CS,CT,CU,CV,CW,CX,CY,CZ=0;
GET DATA (NSTEPS,LABEL,A,B,C,D,CA,CB,CC,CD,CE,CF,CG,CH,CI,CJ,CK,CL,CM,
    CN,CO,CP,CQ,CR,CS,CT,CU,CV,CW,CX,CY,CZ);
T0,U0,V0,W0,X0,Y0,Z0=0; TN,UN,VN,WN,XN,YN,ZN=0;
END RESET;

PRINT:PROCEDURE; /***************************** PRINT A LINE *******/
```

FIGURE 1. MATHPROGRAM IN PL/C, CONFIGURED FOR NEWTON'S METHOD TO SOLVE X + 2 - EXP(X) = 0.

```
P(1)=TNM1; P(2)=UNM1; P(3)=VNM1; P(4)=WNM1; P(5)=XNM1;
P(6)=YNM1; P(7)=ZNM1;
IF LC=LM THEN CALL HEADINGS;  PUT EDIT(N)(SKIP,F(4));
DO J=1 TO NSEQ; Q=P(J); R=ABS(Q);
IF (R<UP & R>=LO) | R=0 THEN PUT EDIT(Q)(F(16,6)); /* PRINT FIXED FORM*/
     ELSE PUT EDIT(Q)(E(16,6));     END;              /* ...OR FLOAT */
LC=LC+1;  END PRINT;                             /* INCREASE LINE COUNT   */

HEADINGS:PROCEDURE;   /* PAGE HEADINGS ***********************************/
PUT  EDIT(LABEL,'CASE',CASE,'NSTEPS',NSTEPS)(PAGE,A,(3)(A(6),F(5),X(3)));
FA:FORMAT(SKIP, (5)(A(2),F(9,3),X(2)) );
PUT EDIT('A',A,'CC',CC,'CI',CI,'CO',CO,'CU',CU)(R(FA));
PUT EDIT('B',B,'CD',CD,'CJ',CJ,'CP',CP,'CV',CV)(R(FA));
PUT EDIT('C',C,'CE',CE,'CK',CK,'CQ',CQ,'CW',CW)(R(FA));
PUT EDIT('D',D,'CF',CF,'CL',CL,'CR',CR,'CX',CX)(R(FA));
PUT EDIT('CA',CA,'CG',CG,'CM',CM,'CS',CS,'CY',CY)(R(FA));
PUT EDIT('CB',CB,'CH',CH,'CN',CN,'CT',CT,'CZ',CZ)(R(FA));
PUT EDIT('N','TN','UN','VN','WN','XN','YN','ZN')
     (SKIP,X(3),A(1),(7)(X(14),A(2) ) );
LC=0;   END HEADINGS;

ON ENDFILE(SYSIN)MORE='NO';                 /************** MAIN PGM STARTS */
CALL RESET;
DO WHILE(MORE='YES');
CALL START;  TNM1=T0; UNM1=U0; VNM1=V0; WNM1=W0; XNM1=X0;
YNM1=Y0; ZNM1=Z0; CALL PRINT;
DO N=1 TO NSTEPS;
CALL ADVANCE; TNM1=TN;UNM1=UN;VNM1=VN;WNM1=WN;XNM1=XN;YNM1=YN;ZNM1=ZN;
CALL PRINT;  END;
CALL RESET;  END;
PUT SKIP(3)LIST('PGM ENDED NORMALLY: NO MORE INPUT DATA'); END SEQUEN;
*DATA
LABEL='SOLUTION OF X+2-EXP(X)=0 BY NEWTONS METHOD'
NSTEPS=10  C=1;
```

FIGURE 1, CONT.

```
SOLUTION OF X+2-EXP(X)=0 BY NEWTONS METHOD                                    CASE      1   NSTEPS    10
A        0.000   CC      0.000   CI      0.000   CO      0.000   CU      0.000
B        0.000   CD      0.000   CJ      0.000   CP      0.000   CV      0.000
C        1.000   CE      0.000   CK      0.000   CQ      0.000   CW      0.000
D        0.000   CF      0.000   CL      0.000   CR      0.000   CX      0.000
CA       0.000   CG      0.000   CM      0.000   CS      0.000   CY      0.000
CB       0.000   CH      0.000   CN      0.000   CT      0.000   CZ      0.000



N
              TN            UN            VN            WN            XN              YN            ZN
0       0.000000      0.000000      0.000000      0.000000      1.000000        0.281718     -1.718282
1       0.000000      0.000000      0.000000      0.000000      1.163953       -0.038616     -2.202569
2       0.000000      0.000000      0.000000      0.000000      1.146421   -4.893375E-04     -2.146911
3       0.000000      0.000000      0.000000      0.000000      1.146193   -8.173544E-08     -2.146193
4       0.000000      0.000000      0.000000      0.000000      1.146193   -2.664535E-15     -2.146193
5       0.000000      0.000000      0.000000      0.000000      1.146193    4.440892E-16     -2.146193
6       0.000000      0.000000      0.000000      0.000000      1.146193    4.440892E-16     -2.146193
7       0.000000      0.000000      0.000000      0.000000      1.146193    4.440892E-16     -2.146193
8       0.000000      0.000000      0.000000      0.000000      1.146193    4.440892E-16     -2.146193
9       0.000000      0.000000      0.000000      0.000000      1.146193    4.440892E-16     -2.146193
10      0.000000      0.000000      0.000000      0.000000      1.146193    4.440892E-16     -2.146193
```

FIGURE 2. OUTPUT FROM THE PROGRAM OF FIGURE 1.

of up to six function names F, F1, F2, G, G1, G2. In the example, F is used for f and F1 for f'.

The data cards in Fig. 1 and the page heading in Fig. 2 illustrate the use of PL/I "data directed" input. The program provides for up to 30 numerical input items, plus a LABEL for the output pages and a value for NSTEPS. The user only has to enter the items actually used. The CASE counter keeps track of multiple sets of data on a single run. The page headings are repeated on each successive page of output.

The program as it appears in Figure 1 is somewhat difficult to read, since we have used multiple-statement lines. This is permissible because the students never have to read the program (except for locating the procedures, which are highlighted), and useful because it holds down the size of the deck to be manipulated (or sorted when it is dropped). We also use PL/I options to suppress output that would not be useful to the students (e.g. attribute and cross reference tables).

Note on line 4A that π and e are provided in the program as PI and E, respectively.

Some other features that we have tried are not illustrated in the figures. For example, it is possible to add a line printer plotting routine to the program, to be triggered by an input item. This would reduce the number of tabulated sequences to two or three and plot their values as functions of N in the remaining space. Our program initially had such a feature, but it was abandoned on the ground that students would learn more by using numerical output to draw their own graphs on coordinate paper.

In the last two sections below we discuss the use of two other versions of MATHPROGRAM. In one, the six functions are converted to functions of two variables X and Y, which is useful for multivariable calculus. In the other, all the arithmetic is done as complex arithmetic, which has obvious advantages for a complex variable course.

We have not implemented an interactive version of MATHPROGRAM, since we don't have enough terminals to provide adequate service in this mode. However, there is no reason why MATHPROGRAM could not be implemented interactively, with instructions to the students to enter the necessary formulas and data.

Freshman Calculus

When the MATHPROGRAM deck is first given to the students, it is configured for tabulating the six functions F, F1,...,G2 on an interval [A,B]. The student enters A, B, NSTEPS, and LABEL on data cards and as many function definitions as the problem requires. (One is provided in the initial deck.) The START and ADVANCE procedures are provided, with START including:

```
D = (B-A)/NSTEPS;
```

and ADVANCE including

```
TN = A + N*D;
```

```
UN = F(TN); VN = F1(TN); etc.
```

We also alter card number 47 for this initial use to print the column headings as N, X, F1, F2, etc. The card with sequence names as column headings is substituted later.

The first three lab sessions (one per week) and corresponding problems use the function tabulation configuration for tabulating polynomial and rational functions (including separate tabulation of numerator and denominator) and their derivatives, computed both analytically and numerically (see [3] or [5, Ch. 32]).

One of the main objectives of the first three problems is to make sure that students gain facility in punching data cards and function cards. To ensure that students develop the habit of studying their output data at an early stage, specific questions are asked as part of the assignment, such as: Graph f. What is f(1.6)? Where (approximately) does the graph of f cross the x-axis? Where (approximately) does f have its largest value on a certain interval?

During the third lab session a short lecture is given on sequences. The emphasis of the lecture is that the n-th term x_n of a sequence can be expressed in terms of n, x_{n-1}, and/or terms of previously computed sequences. These ideas are illustrated with numerous examples and further reinforced with a homework assignment. The purpose of this lecture/assignment is to prepare the students for a discussion of the program as a sequence tabulator.

Assignments four through seven are designed to give students considerable practice in writing START and ADVANCE procedures. They tabulate powers, reciprocals, sums, factorials, alternating sums, and various important combinations of these. Assignments 6 and 7 include compound interest for various periods of compounding [5, Ch. 45] and Newton approximations for square, cube, and fifth roots [5, §4.4].

Experience indicates that after seven lab sessions the students are quite comfortable with the program deck and have mastered the techniques of punching START, ADVANCE, function, and data cards. The significant application of the computer to the solution of calculus problems begins with lesson 8. Typical problems for the remainder of the first semester include finding roots, relative extrema, and inflection points, by Newton's method [5, Ch. 19], and approximating integrals by upper and lower sums [5, Ch. 24] and trapezoidal and midpoint rules [5, Ch. 26].

In the second semester we compute arc length and surface area by the approximations that lead to integrals and then introduce Simpson's rule for several problems [5, Ch. 29]. The ADVANCE formulas for Simpson's rule might be

```
XN = XNM1 + 2*D;

YN = F(XN);

ZN = F(XNM1 + D);

TN = YNM1 + 4*ZN + YN;

UN = UNM1 + TN;

VN = UN*D/3;
```

```
VERIFICATION OF FTC, Y=1/X, 1<=X<=3                                              CASE      1   NSTEPS   20
A       1.000   CC      0.000   CI      0.000   CO      0.000   CU      0.000
B       3.000   CD      0.000   CJ      0.000   CP      0.000   CV      0.000
C       0.000   CE      0.000   CK      0.000   CQ      0.000   CW      0.000
D       0.050   CF      0.000   CL      0.000   CR      0.000   CX      0.000
CA      0.000   CG      0.000   CM      0.000   CS      0.000   CY      0.000
CB      0.000   CH      0.000   CN      0.000   CT      0.000   CZ      0.000


 N          TN           UN          VN          WN             XN             YN             ZN
 0    0.000000     0.000000    0.000000    0.000000       1.000000       1.000000       0.000000
 1    5.718615     5.718615    0.095310    0.953102       1.100000       0.909091       0.952381
 2    5.220685    10.939300    0.182322    0.870114       1.200000       0.833333       0.869565
 3    4.802564    15.741864    0.262364    0.800427       1.300000       0.769231       0.800000
 4    4.446479    20.188343    0.336472    0.741080       1.400000       0.714286       0.740741
 5    4.139573    24.327916    0.405465    0.689929       1.500000       0.666667       0.689655
 6    3.872312    28.200228    0.470004    0.645385       1.600000       0.625000       0.645161
 7    3.637478    31.837706    0.530628    0.606246       1.700000       0.588235       0.606061
 8    3.429505    35.267211    0.587787    0.571584       1.800000       0.555556       0.571429
 9    3.244034    38.511245    0.641854    0.540672       1.900000       0.526316       0.540541
10    3.077598    41.588842    0.693147    0.512933       2.000000       0.500000       0.512821
11    2.927410    44.516252    0.741938    0.487902       2.100000       0.476190       0.487805
12    2.791201    47.307454    0.788458    0.465200       2.200000       0.454545       0.465116
13    2.667106    49.974559    0.832909    0.444518       2.300000       0.434783       0.444444
14    2.553577    52.528136    0.875469    0.425596       2.400000       0.416667       0.425532
15    2.449320    54.977456    0.916291    0.408220       2.500000       0.400000       0.408163
16    2.353243    57.330699    0.955512    0.392207       2.600000       0.384615       0.392157
17    2.264420    59.595119    0.993252    0.377403       2.700000       0.370370       0.377358
18    2.182059    61.777177    1.029620    0.363676       2.800000       0.357143       0.363636
19    2.105479    63.882656    1.064711    0.350913       2.900000       0.344828       0.350877
20    2.034093    65.916750    1.098612    0.339016       3.000000       0.333333       0.338983
```

FIGURE 3. OUTPUT FROM MATHPROGRAM SET UP TO TABULATE AN INTEGRAL BY SIMPSON'S RULE (VN) AND DIFFERENTIATE THE RESULT (WN). COMPARE WN AND ZN COLUMNS.

The student chooses NSTEPS, the number of double subintervals. The last entry in the VN column is the answer.

Moreover, the VN column is a tabulation of an (approximate) antiderivative of F. Thus it is possible to provide a dramatic illustration of the Fundamental Theorem of Calculus by adding one more advancing formula to differentiate this tabulated function numerically:

```
WN = (VN - VNM1)/(2*D);
```

This approximates the derivative of V at the midpoint XNM1 + D, which may be compared with ZN (Figure 3).

Other problems in the second semester include solution of initial value problems by Euler's method [5, Ch. 48], computation of transcendental functions by polynomial approximations [5, Chs. 39, 37, 42] and evaluation of series [5, Chs. 52, 54, 56]. A complete set of these problems, including many alternates and variations, will be included in the package to be submitted to CONDUIT.

Multivariable Calculus

During the fall term of 1977 the computer was used by students in one section of our third semester calculus course. A few of these students had used MATHPROGRAM as freshmen, most of the others had completed a programming course in either the Computer Science Department (PL/C) or the Engineering School (FORTRAN), and only a few had no prior computer experience. These last few were given a demonstration of keypunching and of running the MATHPROGRAM deck and interpreting the output. Otherwise, all computer-related instruction was done by written handouts. There were no extra lab sessions, and nothing was removed from the standard content of the course (but a lesson on numerical integration was added.)

The computer work was done with three different decks. The first was the same program described above, but initially configured to tabulate coordinates of points on a curve defined parametrically (just a multiple function tabulation). The second was a three-dimensional graphics program called SURFACE, described elsewhere [6], [7]. The third, which we call MATHPROGRAM II, differs from MATHPROGRAM only in that the six functions are made functions of two variables. In all, 18 problems were assigned, timed to mesh with the content and pace of the course (roughly chapters 12-15 and 18 of [9]), not on a regular schedule. Two to three weeks were allowed for completion of each problem, and "credit" was given for this part of the course on completion of at least eight problems.

The first group of problems included tabulation and graphing of vector valued functions (plane and space curves defined parametrically) and their derivatives, arc length of a space curve via Simpson's rule, and solution of the predator-prey equations by Euler's method [5, Ch. 50], with both the individual populations and the trajectory of solutions to be graphed.

The initial configuration of the MATHPROGRAM II deck included a tabulation of a function of two variables over a rectangle $a \leq x \leq b$, $c \leq y \leq d$ in its domain. The following START and

and ADVANCE formulas were provided without explanation:

```
     START:

          CD = (B-A)/CX + 1;

          NSTEPS = FLOOR(CD*((D-C)/CY + 1)+ .5);

     ADVANCE:

          XN = A + MOD(N-1, FLOOR(CD + .5))*CX;

          YN = C + FLOOR((N- .5)/CD)*CY;

          ZN = F(XN, YN);
```

The user enters only the end points A, B, C, D, and the step sizes CX(= Δx) and CY(= Δy) as data .

Later this same scheme for "walking" a gridwork on a rectangle became the basis for evaluating double integrals by Simpson's rule in two variables [1, pp. 511-516]. This also permits evaluation of integrals over circular sectors by conversion to polar coordinates.

Another group of problems used MATHPROGRAM II to find simultaneous solutions of equations in two variables by Newton's method [9, p. 712, Prob. 7], with application to relative extrema and saddle points. Classification via the second derivative test is easily accomplished by tabulating the discriminant along with the approximating sequences for the critical points.

Finally, we were able to do least squares fitting to real data (linear regression) [9, Section 14-10] by using an input statement in the ADVANCE procedure and computing the coefficients of the normal equations:

```
GET LIST (XN, YN);

TN = TNM1 + XN;

UN = UNM1 + XN**2;

VN = VNM1 + YN;

WN = WNM1 + XN*YN;
```

The rest of the solution was obtained by calculator. (Some statistical calculators have hardware for the whole job, but sophomore calculus students don't know what to do with it.)

Other Uses of MATHPROGRAM

A version of MATHPROGRAM using complex numbers and complex arithmetic has been used in an introductory complex variable course for undergraduates. The use of the program permits a tremendously expanded experience with complex numbers, which is normally not available to students. One can deal with non-holomorphic maps, i.e. those given by $w = f(z,\bar{z})$, procedures for locating roots of such functions, including an appropriate version of Newton's Method, and procedures for expressing Riemann surfaces in precise detail which are very informative.

The basic characteristic of a function of a complex variable is that of a map of the complex plane into itself. This is most effectively elucidated by obtaining the images of closed curves, which is one of the capabilities of the program. Zeros can be located by determining whether such an image encircles the origin. Stability criteria such as the Nyquist Criterion are readily available. Riemann surfaces are usually formulated in terms of non-intersecting "slits" which are rays originating at the image of branch points. Such a ray will usually have two pre-images which join at the branch point. A suitable Newton procedure will obtain these preimages, which divide the domain plane into regions, each of which is mapped into a "sheet" of the Riemann surface. The connectivity of this mapping is readily explored by mapping appropriate curves.

A sequential program obviously permits one to realize step by step procedures for solving initial value problems in ordinary differential equations. The use of Euler's Method has already been noted above. By using the parameters (CA, CB, etc.) as intermediate quantities, one can implement Runge-Kutta methods. The parameters can also be used to store past values of the derivatives from one advance step to the next and permit predictor or predictor-corrector procedures, provided one sets up the appropriate initial situation. This can be obtained by going backwards by the Euler method with a smaller step size.

However, packaged programs are usually available for these methods and one is only concerned with relatively simple procedures as introductions to the subject. The advantage of a fixed format is that the instructor can obtain considerable inspiration and understanding by exploring various possibilities without the customary programming impedimenta.

Finally, we might note the use of MATHPROGRAM as a handy research tool. If one wants to know, say, the sum of a series or the behavior of a differential system, it is easier to use MATHPROGRAM than to use a programmable calculator, and (assuming a nearby keypunch and batch terminal) only slightly less convenient.

Notes

[1] Department of Biomedical Engineering, Duke University.

[2] Department of Computer Science, Duke University.

[3] Presenter of paper. Direct all correspondence to Dr. Smith.

References

1. Flanders, H., Korfhage, R. R., and Price, J. J., A Second Course in Calculus, Academic Press, New York, 1974.
2. Garland, S. J. (Ed.), Calculus and Computing, American Mathematical Society and Mathematical Association of America, 1975.
3. Smith, D. A., "Numerical differentiation for calculus students", Amer. Math. Monthly 82 (1975), 284-287.

4. Smith, D. A., "Computer calculus without programming", unpublished. Presented to the American Mathematical Society under a different title, January 24, 1976. Abstract 731-98-1, *Notices Amer. Math. Soc.* 23 (1976), A-226.

5. Smith, D. A., *Interface: Calculus and the Computer*, Houghton Mifflin, Boston, 1976. Also available as CONDUIT package MTH 105, 1977.

6. Smith, D. A., "Inexpensive computer graphing of surfaces", *Math. Mag.* 50 (1977), 143-147.

7. Smith, D. A., *SURFACE: A Three-Dimensional Graphics Module for Multivariable Calculus*, CONDUIT, 1978.

8. Stein, S. K., *Calculus and Analytic Geometry*, 2nd ed., McGraw-Hill, New York, 1977.

9. Thomas, G. B., Jr., *Calculus and Analytic Geometry*, Alt. ed., Addison-Wesley, Reading, 1972.

INTERWEAVING MATHEMATICS AND COMPUTERS FOR LIBERAL ARTS STUDENTS

Kenneth John Preskenis

Framingham State College

This article presents the highlights of a successful attempt to combine computers and mathematics in a course in order to alleviate mathematical avoidance and/or alienation in college students whose overall high school performances were above average but whose mathematical performances were average or below. The work was done at Boston College in the fall of 1976 with fifteen students and the computer support consisted of a fine student-oriented computer center containing an IBM 370 computer with time sharing capability for a large number of terminals. However, based on the author's experience in both a small college and a large university the value of the work could have been retained at a school which had merely one terminal linked over a telephone line to a smaller computer such as a Hewlett Packard 2000. Further, it is hoped that the value could be retained with a larger group (up to, perhaps, 35) and with non-college students.

The externalization of this work was a one semester course with many common instructional procedures. In particular, class sessions consisted of a combination of lecture and workshop. A text was specified but used as a reference book; teacher-constructed handouts were distributed at most sessions in an attempt to minimize note taking in class and to increase the teacher's involvement with the material. The emphasis throughout the course was on inquiry and clear thinking rather than instruction and practice. The course was broken up into three distinct components. Evaluation of each student's performance was based on three homework assignments in each component, three component exams, a comprehensive final exam and a project from an area of interest to the student. The exams were comprised of both objective and essay-type questions and the format of the exams came from classroom discussions. Sample projects were a six page term paper and a computer program which gave the user a five question multiple choice quiz.

One uncommon feature of this enterprise was that an unusual amount of time and energy was spent in designing and preparing the presentation of this course. (This work was partially supported by a two year faculty fellowship at Boston College.) For instance, a major effort went towards developing, formulating and refining a concept of the course which would "fit in with" the category of students scheduled to be in the course. The resulting notion was that of an introductory course in mathematical and computer modeling. The line of thought supporting this notion was as follows: the scientific process of "modeling to analysis to computing" seems to exhibit not only mathematics but also how it is used and thereby would convey to students the usefulness of what they are

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

studying. Further effort went towards trying to understand and appreciate what would be most beneficial for the students involved and thereby reach a formulation of a primary purpose for the course. Since this one semester course could very well be a terminal mathematics course for many of the students, it was decided that to concentrate on the problem-solving ability of the students or to seek the mastery of mathematical topics by the students was inappropriate. Since the students would have above average ability in some areas other than mathematics, a more legitimate (and at the same time, realistic) purpose was the development in the student of some understanding of and some real attitude towards mathematical ideas. Of course, this development would be constrained in a natural way because of the students' limited disposition and capacity for mathematics. However, there was a conviction that the mathematical ideas could be illustrated, illuminated, and conveyed through the "vehicle" of carefully selected mathematical models in the sense that the models would not be treated as ends in themselves but rather as instruments to aid in the exposition of the ideas. This conviction was partially based on the belief that most thinking people deal with problems in the real world by generating a mental model of those problems and to the extent that this belief was true for the students concerned, the process of generating a non-mathematical abstract model to deal with a problem could be used as a starting point and constant reference point throughout the course. Of course, "manufacturing" a mathematical model and analyzing that model on a purely mathematical level would be foreign to the students and compensation would definitely be needed to bring the mathematics over to a more meaningful level. The basic strategy would be to start with the knowledge and at the level of communication that the students had already acquired (two years of algebra and one year of geometry were prerequisites; and a week and a half was devoted to reviewing relevant parts of this material) then to present to the students some simplified non-contrived problems in which technical details are held to a minimum; and finally to lead the students through the solution of these problems using the "modeling to analysis to computing" process. Concerning the latter step, the teacher would have to demonstrate extraordinary patience in breaking down various mathematical ideas to step-by-step tasks and in constantly spelling out to the students how these tasks fit together to reflect a deeper, more abstract thing - an idea. Computer oriented algorithms would complement this part of the teacher's work. Another important ingredient in leading these students would be to get them to develop an attitude towards the ideas by exercising their judgement with respect to aspects of those ideas. For this, the teacher would need not only patience but openness and flexibility in handling the students' attempts at making judgements about aspects of mathematical ideas. And for this, certain computer programs were chosen which would require the students to exercise their judgement while using the program. Thus, the computer operating in its algorithmic request-response way and supplying a user generated output sheet would play an important role in "slowing down" the mathematical analysis and in helping students make and confirm judgements. Furthermore, if the uses of the computer were made to be quite different, then a valuable by-product of the course would be the implanting in the students of a feeling for how the computer is used and what are its limitations.

The next stage prior to actually implementing the course was the professional activity of screening different mathematical models and possible presentations of them to see what would truly serve the concept and purpose of the course. It almost goes without saying that this involved much professional reading and discussion together with trying to weave the concept and purpose into whatever seemed to be relevant. It was eventually decided to exhibit the "modeling to analysis to computing" process in three distinct areas; for problems which draw from linear programming; for problems which draw from statistics; and for problems which draw from calculus. Also since the computer was to be used in each of the three components, instruction in the use of it would be provided "along the way" for those students who were unfamiliar with it. It should be mentioned that no knowledge of a programming language was required in any of the computer uses. Also it should be emphasized that the computer uses were integrated into a broader instructional setting, namely, the students coming to grips with a few important ideas in mathematics and, hopefully, acquiring some feeling for how these ideas are used in the three areas of business, social science and science.

As the presentation of the course began, two realities seemed to take center stage: first, the commitment on the teacher's part to implement the the course with its concept, purpose(s), organization of topics, and instructional procedures; second, the adoption of an atmosphere of experimentation by both teacher and students. Concerning the commitment, there was a strong belief by the teacher that the course would benefit all concerned and a disposition to carry this belief to all parts of the course. Concerning the second reality, it seemed important that both teacher and student accept that the course would be a human enterprise, that it would be subject to errors in judgement and thus, it was legitimate to expect experiences of failure as well as success. It seemed helpful for teacher and students to know that if any defensible teaching-learning strategy arose which was at odds with existing regulations, then the regulations would be altered. On the other hand, a systematic approach (three class sessions per week with homework and exams as indicated earlier) to insure quality and standards had the consent of all concerned.

Since the students had real barriers to accomodating mathematical information, seemingly minor details sometimes led to major events. For instance, the designated classroom had one small portable blackboard which seemed awkward for presenting mathematics. However, the sparse blackboard space dictated far more student-teacher dialogue than normal for a mathematics course. Also the classroom was across the hall from the computer terminals and, several times after class, there was a convenient walk to a terminal to correct a student's misunderstanding. The point here was that some learning took place in "another" room outside of the classroom. In order to orient students and to spark the activity of using their own judgements, the first assignment was to produce six pages of quotes by famous people on either the learning of mathematics or the teaching of mathematics. The quotes in each paper served as common ground for discussion between teacher and student throughout the course.

The first instructional unit lasted about three weeks and covered the solution to linear programming models in two and three dimensions. The emphasis was on the geometric approach; the simplex method and matrices were omitted. There was precisely one practical point of view, namely, all problems were concerned with minimizing or maximizing profits as a management decision in the context of the oil industry. This component began by distinguishing between numbers as first degree abstractions and variables as second degree abstractions (thinking about thought). This put into a broader context certain high school algebra techniques including solving equations of the form $ax + b = 0$. The concepts and language of set theory and logic were taken in stride (as they came up). The "classical" model consisted of a finite number of inequalities involving linear expressions in two or three variables and required being able to solve n equations in n unknowns where $n = 2$ or 3. The students learned how to translate specfic word problems into this classical model. Enough analytic geometry in two and three dimensional space was presented so that the students could construct polygons representing "pictures" of feasible solution sets. It seemed important that each constant and each variable measured some specfic item and, thus, had some concrete meaning. Next, about two weeks were spent on the analysis of these linear programming models including the discovery of how a linear objective function (a third degree abstraction) takes its minimum or maximum at the vertices of previously constructed polygons. Computing entered through two "service programs" (one for two dimensions and one for three dimensions) which were quick and straightforward to use. In fact, in order to use them the student simply had to press a couple of buttons on the terminal, to type in an identification number, the word "run" and the data which represented the constraints and the objective function. The algorithm and accompanying flow chart (it was essentially the same for each program) were presented in complete detail during class sessions so the students would understand the program. The computer output represented a nice practical counterpart to the theoretical solution.

The specific benefits which accrued to these mentally able but mathematically alienated students from using the computer were as follows: (i) the programs required (and so fostered) mathematical thinking; (ii) the programs took the drudgery out of computation; (iii) the output of the programs confirmed the mathematical idea of a linear function taking its extreme values at the vertices of a polygon. In particular, the computer was a link between a mathematical model and the problem which spawned it.

The second component of the course lasted five weeks and covered the data gathering and descriptive techniques of statistics and certain probability models which were compatible with collected data and which helped interpret the data. The fact that collected data led to certain general models gave the students a view of model building "from scratch" which was not conveyed in the initial component. The first assignment required each student to have fifteen simple questionaires filled out by a sample of their choosing. The students' results led to class discussions about the samples and to conjectures about the Boston College population as a whole e.g., what is the proportion of males at Boston College in view of the specific results in the sample. Again numbers and variables were the starting point, however, in this setting some variables were devices for quantifying nonnumeric objects such as human qualities or attitudes. Also graphing appeared again as an important ingredient in expressing initial observations and in analyzing possible interpretations. During the week and a half of descriptive statistics a computer program supplying the exact mean and standard deviation was used together with a careful comparison to the hand calculated approximations which came from grouping data and using so called class marks. In the two weeks that followed, probability models of coin and dice tossing experiments were presented with emphasis on the tie between equally likely physical outcomes and the psychological notion of equally likely. Class sessions cultivated the idea that the theoretical probabilities agreed with intuitive projections of those probabilities. There was a different computer use here, namely, a computer simulation program for randomly drawing cards from a container. With this program the student could calculate the "empirical" probability of an event and compare it to the theoretical probability of the same event in the probability model. Finally, it was decided to present enough ideas from correlation and regression to get more mileage from the student questionaires and also to construct the linear forecasting model which ties the logarithm of the population of the U.S.A. to the years from 1890 to 1950. One simple number crunching program was used to supply the correlation coefficient and the slope of the regression line equation in response to student-submitted sample data. Although the algorithm for this program was not explained in class, the students did have to calculate these statistics by hand for simple data several times.

A specific benefit of this second component was that the students acquired a deeper understanding of the point of view of applied mathematics. In particular, there were problems which seemed to be outside of mathematics - such as how is a person's political attitude related to the attitude about the legalization of marijuana - which after some working hypothesis could be cast into a linear regression model. And wherever the results of the model agreed with our experience, the model could be used as a working model. The computer contributed to the construction of the model by supplying calculations quickly and by allowing a crude model to be refined (or if necessary, scrapped in favor of an alternate approach).

The third component of the course lasted three weeks. It presented the ideas of function and derivative in terms of the physical setting of elementary motion problems and then in completely general terms along with several other illustrative examples. One week was spent on motion problems using the intuitive approach of W.W. Sawyer (see 3). Next there was much traveling back and forth between the context of motion where the variables t and s had meaning and the more abstract setting where x and y were pure variables which were functionally related. Computer help was available here through a fine computer program from Hewlett Packard (see 2) which was designed to be run by the student but not listed, i.e., the student could not look at the program, only execute it. The point of the program was to have the student input values of x (perhaps, in some monotone order to give the feeling of a variable) and receive the values of an "unknown" function (and its derivatives as an option of the teacher) in return. Since the students were confronted by these functions in their purest form, i.e., as a table of values, rather than in formula form, they appreciated the general definition of function a little better. Also the teacher had the students exercise their judgement while practicing the activity of graphing by putting many different functions including complicated rational functions into the program. Specifically, the students could use the program to graph functions by the important method of plotting a few points, guessing the curve between the plotted points, checking the guess by plotting a couple of intermediate points and repeating the process, if necessary. This method is unrealistic when employed by hand on a complicated function. For more information on this component, see 1.

Looking back objectively at the presentation of the course it is hard to verbalize much of the day-to-day presentation and it is not at all clear how much of it would be transferable to other situations. However, a few comments may be useful depending on the teacher, the students and the classroom context. These will now be expressed.

1) There was a real problem of a teacher with elaborate mathematical training being foreign to the mathematically alienated students. Perhaps, a teacher should constantly contemplate personal experiences of being a learner and should convey something of these experiences to the students. This would show human concern for the students and their "position of awkwardness". With due respect to the fact that the teacher showing too much interest can be a distraction for the student, these math-alienated students seemed to be able to assimilate mathematical information and to put it far away in a mental compartment; further, they may or may not be able to (or choose to) open this compartment again later. It could just be that a positive image of a "human" teacher is enough to make some students practice opening these closed mental compartments and accomodating the information inside.

2) It was helpful in the day-to-day presentation to draw out reactions and questions from the students, not only to find out "where the students are", but also to shed light on the mathematical points by supplying different (students) points of view to the (other) students. Devices which do this are not readily available in mathematics courses; although the great mathematician and teacher Polya once remarked ...'if there is a mathematical statement on the board, then the teacher can always take a vote to see whether it is perceived as correct'... Perhaps related to this, most class days started off with a quote by a famous person from former times and a brief discussion of how that quote related to present times.

3) The students seemed interested in recent articles that the teacher had read. For instance, discussing in class the Time magazine article which reported that a famous psychologist falsified his correlation statistics on the heredity of I.Q. made a significant impression on some students.

The author would like to close with a summary of the evaluative information on the presentation of the course. First, informal comments made by the students to the director of their overall program, Professor A. Folkard, indicated "the course was very successful". To the extent that this is accurate, the author would like to acknowledge the perceptive but non-imposing support of both Prof. Folkard and Prof. G. Bilodeau, chairman of the department of mathematics. Also the author owes a debt to two former students, P. Cunningham and M. Griffin, for beneficial conversations about the design of the course. Eighty percent of the collection of so called course evaluation questionaires filled out voluntarily by the students rated the course highly. The paper concludes with some selected quotes from these questionaires.

"It is a new approach to math and has given me more interest in math as a science not a set of calculations."

"The idea and organization of the course are great!"

"The content was new and interesting, and it was really quite practical - working with computers was a welcome facet of the course I had not expected."

BIBLIOGRAPHY

1. "Computer Assisted Instruction in Differential Calculus", Preskenis, K. J. (Proceedings of 1976 Conference on Computers in Undergraduate Curricula. SUNY at Binghamton; pgs. 229-232)

2. "Project SOLO Computer Topics-Mathematics Projects", (A publication based on materials produced in a joint effort of the University of Pittsburgh and the Pittsburgh Public School System under the directorship of Dr. Thomas A. Dwyer with copyright by HEWLETT PACKARD COMPANY of Cupertino, California)

3. "What is Calculus about?", Sawyer, W. W. (A book-from a series called New Mathematical Library-with copyright by Yale University and with publication by Random House, Inc.)

COMPUTER AUGMENTED VIDEO EDUCATION (CAVE) IN ECONOMICS

by

Rae Jean B. Goodman
Economics Department
U.S. Naval Academy
Annapolis, Maryland 21402

Basic Economics (FE 210), an economics principles course, is taught at the U. S. Naval Academy to 85% of the student body. Approximately 15% perform unsatisfactorily in the course each semester, and instructors expend large amounts of time giving extra instruction, repeatedly, over the same topics. During the late sixties and early seventies, a variety of approaches were tried by several economics faculty members including interactive computer games and educational television to teach specific topics in Basic Economics. These efforts were uncoordinated and sporadic, at best. Given this historical background, we enthusiastically joined in the development of the Computer Augmented Video Education (CAVE) Project.

The objective of the CAVE project is to combine the advantages of the computer with the capabilities of educational television to form an effective instructional tool. Since the Naval Academy already maintained the videocassette playback units, the CAVE project members decided to use a computer terminal which "linked up" with the playback unit using the same video monitor. Thus, the student may employ either the computer mode or the videotape mode, but not both simultaneously. The student moves between the two modes by manually pressing a transfer switch. This set-up allows for two fundamental interaction sequences as depicted in Figure 1. In A, for example, the user views the videotape, stops it at the desired point while automatically moving to the computer mode to self-test the understanding gained. In B, the user views a portion of the videotape, moves to the computer mode, than back to the interrupted point of the videotape for further educational instruction. The B sequence may be repeated any number of times during a specific videotape because the transfer switch is manually operated.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

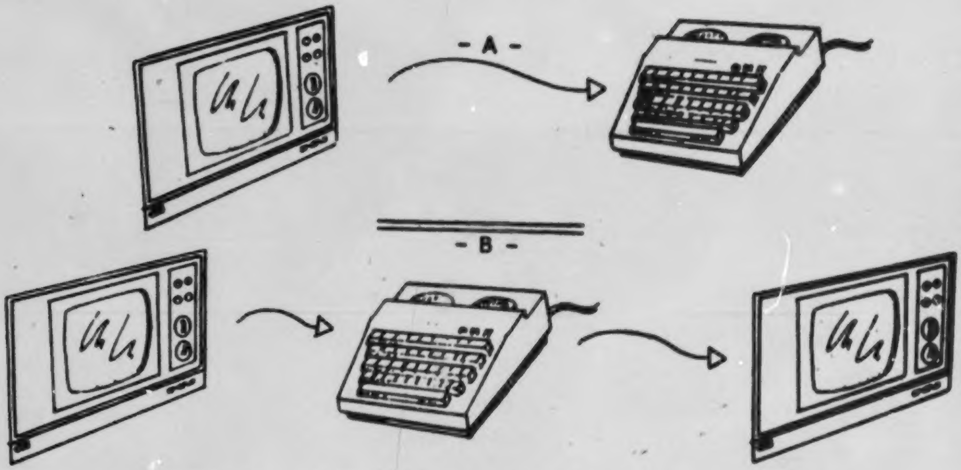


Figure 1: **BASIC INTERACTION SEQUENCES**

The CAVE programs allow for a large amount of student control. The author of the program includes suggested points of transfer in both the television and computer portions. However, the student determines the actual physical interaction between the videotape and the computer program by means of the transfer switch.

The computer programming staff developed a highly flexible instructional program for the computer portion of the CAVE project. The program allows for (1) various teaching strategies, (2) student help sequences, (3) student-selected moves through the program, and (4) a variety of answer types.[1] All of the options allow for a flexible program format and prevent boredom for repetitive users of the programs.

The goal of the Economics Department is to create a series of programs that cover the major areas in the Basic Economics course. The pilot project in Economics developed six CAVE programs in the area of macroeconomics entitled: Keynesian Analysis, Income Expenditure Multiplier, Fiscal Policy, Bank's Balance Sheet, Money Creation, and Monetary Policy. A description of each of the six programs appears in Table 1. The basic format of each videotape is given in Figure 2. The introduction and conclusions were done with the instructor in the studio, in an office, or on location. The different backgrounds were used to add some variety to the programs. For example, the introductions and conclusions of the two programs dealing with commercial banks were videotaped in front of a local commercial bank. Similarly, to maintain interest, the audio component of the videotape was done by either the instructor or a professional announcer. The duration of each videotape ranges from ten to twenty minutes. On the short tapes, the first interaction scheme, as shown in part A of Figure 1, is followed; on the longer tapes, the second scheme, part B of Figure 1, is employed.

---

[1] More information concerning the computer portion of CAVE is available on request.

TABLE I

DESCRIPTION OF ECONOMICS CAVE PROGRAMS

FILE NAME	TITLE	DESCRIPTION
FE9504	KEYNESIAN ANALYSIS	The video portion introduces the student to the two-sector and three-sector Keynesian models. Both a graphical and a functional approach are explored and both an aggregate supply and demand approach and a savings-investment equilibrium approach are explained. Taxes and transfer payments are included in the model. In the computer portion, the student works with both a two-sector and a three-sector model. The questions start at a very elementary level to test the understanding of the model's fundamental components; the questions then increase in difficulty.
FE9508	INCOME EXPENDITURE MULTIPLIER	The videotape shows the student the effect of changes in investment and changes in government spending on the equilibrium level of income and investigates the balanced budget multiplier. The model used is the three-sector Keynesian model. The computer portion involves two models--one with autonomous investment and one with induced investment. The questions focus on the impact on equilibrium level of income of changes in government spending and investment by business.
FE9509	FISCAL POLICY	The video and computer portions examine the impact on the equilibrium level of income of changes in government spending, taxes and transfer payments in the three-sector Keynesian model with autonomous and induced investment functions.
FE9510	BANK'S BALANCE SHEET	The video and computer programs explore the various items on a simplified balance sheet of a commercial bank and the impact of transactions on the balance sheet.
FE9511	MONEY CREATION	The program examines how a commercial bank expands the money supply and the limits on the expansion. The explanation is given in balance sheet terms.
FE9516	MONETARY POLICY	The program explores the various policy tools available to the Federal Reserve and the impact each of the tools has on the commercial banking system and the economy.

I	INTRODUCTION — INSTRUCTOR - - - - -	studio, office, or "on location"
II	INSTRUCTIONAL MATERIAL USING VISUALS -	audio by Instructor or "professional voice"
III	SUMMARY AND DIRECTION TO COMPUTER INSTRUCTOR - - - - - - -	studio, office, or "on location"

Figure 2. BASIC FORMAT OF VIDEOTAPES

The six economics CAVE programs cover material that is taught in the classroom and/or is available in the textbook. The videotapes are made available to students who may not have comprehended the classroom instruction or who were absent the day of the classroom presentation. The computer portions allow the student to practice and assure her(him)self of understanding the material. The programs were used on a voluntary basis during the fall semester 1977. Specific information was collected each time a student used the computer portion, covering the following:

1. Student identification number, date and time.
2. Length of time on program.
3. Number of correct answers (on first, second or third attempt).
4. Number of incorrect answers.
5. Student's answer to incorrectly answered questions and the correct answer.
6. Student comments.

During the pilot study 7.5% of the 529 students enrolled in Basic Economics used a CAVE program; and each student used the CAVE programs an average of four times. At the time of completing the computer portion of each program, the student was asked to comment. These comments were mixed, as can be seen in sample comments listed below (edited only for spelling and format). The direct (person-to-person) student response was very positive.

THIS IS A VERY GOOD LEARNING TECHNIQUE.

WHEN I FIRST GOT ON IT WAS A HASSLE BECAUSE I DIDN'T KNOW MY SECTION NUMBER. THEN AFTER FUDGING UP THE FIRST ATTEMPT I LEARNED IT IS ADVANTAGEOUS TO LOOK AT THE TAPE BEFORE TRYING THE PROBLEM.

I THINK THAT THE FIRST QUESTION WOULD HAVE BEEN BETTER IF IT WAS MULTIPLE CHOICE. ALSO, THERE WERE NO INSTRUCTIONS FOR USING THE COMPUTER ON THIS TAPE.

ALL ANSWERS THAT ARE NOT STRICTLY NUMERICAL SHOULD BE OFFERED BY MULTIPLE CHOICE.

MY MAJOR COMPLAINT IS THAT WHEN YOU DON'T KNOW THE ANSWER, THE PROGRAM FLASHES THE SOLUTION BY YOU SO FAST THAT YOU DON'T HAVE TIME TO READ IT OR UNDERSTAND IT.

I COULD NOT SIGN ON THE COMPUTER'S CALCULATOR MODE.

MAKE QUESTION MORE CLEAR TO THE STUDENT.

VERY GOOD PRACTICE EXERCISE. IT ENABLES THE STUDENT TO BETTER UNDERSTAND THE ECONOMIC CONCEPTS STUDIES IN CLASS AND THROUGH THE TEXTBOOK. IT HAS REALLY HELPED ME. THIS EXTRA WORK SHOULD BE CONSIDERED BY THE RESPECTIVE PROFESSORS WHENEVER ASSIGNING A STUDENT A LETTER GRADE FOR A PARTICULAR COURSE.

IT IS ADVISED TO USE FILM FIRST.

A mid-term exam was administered to all Basic Economics students in mid-November. The performance on a specific exam problem which represented the material covered on the CAVE programs was collected along with CAVE usage as a preliminary indication of the effectiveness of the CAVE approach. The results may be found in Table 2. (The discrepancy of sample size and total population size is due to the fact that not all faculty transcribed the necessary data.)

TABLE 2

RESULTS OF PRELIMINARY TESTING OF CAVE

Variable	Mean	Std. Dev.	N	t
Users	24.083	5.897	24	1.5719*
Non-users	21.933	6.464	270	

* Significant at the .058 level.

Our null hypothesis is that the mean scores of the CAVE users and non-users are equal; the alternative hypothesis is that the mean score of CAVE users is greater than the mean score of non-users. The calculated t-value is 1.5719, while the theoretical value at a 10% level of significance is 1.282. Therefore, we may reject the null hypothesis that the mean scores of the two groups are equal.

We have shown in our initial testing of the six economics CAVE programs that CAVE users perform statistically significantly better than non-users on a specific exam problem. The student response was also very favorable. Given these results, the economics faculty member involved in CAVE intends to complete a series of CAVE programs covering the major topics in the Basic Economics course. Further testing of the CAVE programs will be carried out during the Spring Semester.

I would like to thank J. C. Thompson and his Educational Resources Center Staff and Richard A. Pollak and James Schwab of the Academic Computing Center Staff for their outstanding and irreplaceable efforts to ensure the success of the CAVE Project.

CAVE — COMPUTER AUGMENTED VIDEO EDUCATION FOR BASIC PHYSICS

by

Billie J. Graham
Physics Department
U. S. Naval Academy
Annapolis, Maryland 21402

Introduction: Both time-sharing computer facilities and video cassettes have a high potential for augmenting the teaching of physics, especially the introductory-level course. This study was undertaken to determine how these two media could be merged and to determine if the resulting programs could effectively teach certain aspects of the basic physics lab.

An inordinate amount of time is required to teach students the techniques of organizing and using data taken in the laboratory. Although, in the chemistry course which is prerequisite to introductory physics, the students have already been exposed to preparing and analyzing graphs, to handling significant figures, and to making measurements, it still seems that a great deal of effort is used continually throughout the semester to teach these concepts effectively. Time in the basic physics laboratory would be better spent if more emphasis could be placed on illustrating the fundamental concepts of physics. The CAVE program on lab techniques is designed to provide students entering the basic physics course either the background for handling laboratory data or a review when it is needed.

Approach: Since the TV-computer programs provide a one-on-one situation with each student, an attention span of 12 to 20 minutes was maintained as a reasonable length for the cassette portion of the program, with the total interactive computer portion being approximately the same length. The TV program attracts the student's attention because someone with whom he is generally familiar, hopefully his instructor, is pictorially explaining a concept directly to him. The computer program keeps his attention by asking that he think through questions and problems and then respond directly to them. By programming the entire flow of ideas to be presented to a student, the pictorial concepts from the video cassette and the written concepts from the time-sharing system become one integrated program.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

The computer programs used to generate comments, questions, replies, and statistics on replies were titled QUEST and QBUILD. These programs, structured to allow a dialogue of comments or questions and replies to be conveniently constructed by an instructor, were developed for this project by Richard Pollak and Jim Schwab of the Academic Computing Center at the Naval Academy. An advantage of these programs is their adaptability to a wide variety of applications. Without going into the details of the programs, an idea of their versatility can be seen by the types of responses allowed for questions. Each index number represents a position in a programming statement which controls the analysis of a question frame.

Index	Description
1	Frame type (major = 1, minor = 0)
2	Answer type 1 = numeric % 2 = numeric range 3 = keyword exact 4 = keyword (or) 5 = keyword (and) 6 = keyword (order) 7 = comment 8 = 2-10 way branch 9 = survey trap
3	Number of allowed attempts
4	Graphics (on = 1, off = 0)
5	Strategy code (1-3)
6	Number of times a frame is to be used with different variables
7	Correct matches for successful branch
8	Next frame - successful
9	Next frame - unsuccessful

Although the programs maintain such statistics as student identification, date, correct answers, attempts, and elapsed time, the programs used to augment the physics lab instruction did not require most of this information. Since the purpose of these programs was to teach some of the important concepts used in the lab, the dialogue developed was largely tutorial in character. Questions were asked, and then students were "guided" to a correct answer as they answered or attempted to answer them.

A unique programming approach arises when the videocassettes are used with time-sharing computer programs. Since the counter number on the videocassette player advances along with the tape, and since the tapes can be easily moved forward or backward, the videocassette tapes can be programmed somewhat like a computer program. The video tape itself can be used to tell the viewer where to "branch" next, or the associated computer program can suggest branching to or from a specific counter number on the videocassette player. Branching in this case may be either a branch from the videocassette to some portion of the computer program or to a different place on the videocassette. Similarly, the computer program may branch to a portion of the program; or, depending on the response to a question, it may ask the user to branch to a specific location on the videocassette tape.

A simplified portion of the flowchart scheme is shown in Figure 1 which indicates how a videocassette tape is integrated into the flowchart. If a student has little or no trouble with the question at X, then a portion of the video tape is skipped and his progress is hastened by sending him to question Z. In this specific tape on lab techniques, questions are asked on the interpretation of a graph. If two of the four questions are missed, it is suggested that a different portion of the video tape be seen for more review. This becomes a case where, instead of generating a new set of computer data for a given problem, an entirely new problem is pictorially presented. There is a minor disadvantage here in that only a limited number of problems can be presented on the video tape, while computer generated problems can create an unlimited number of problems for each student and for each time he attempts it.

Each TV-computer program is completed individually by a student at his own pace; and this means that unless very strict programming limits are set, he can go through the program in his own fashion. For this reason, a choice is always given to the user as to whether he wants to return to the video tape or to continue with the computer program without viewing the tape at a logical branching point. For example, a student might want to continue with the computer program without going back to the video tape if he were reviewing or if this were his second time through a given tape.

The purpose of each program is to prepare students to better understand either the concepts involved or techniques to be used in specific lab experiments. One of the programs produced developed the equations for projectile motion and then reinforced the ideas developed by helping the user work his way through a series of questions based on these equations.

A program, which is designed to be used prior to beginning the labs, illustrates how uncertainties in measurements appear, how to organize experimentally collected data, how to construct and analyze graphs, and how to handle significant figures. As each of these topics is completed, the program branches and students are questioned by the computer about a given topic before they continue. The questions are designed to make the students think about what she/he has just seen. The responses to these answers are designed to reinforce the correct answers or to help the person arrive at the correct answer, and then send him/her on.

Plans have been made to produce other programs for labs which will smooth out rough spots before a student gets to the experiment itself. These include programs on such topics as accelerated motion, energy, and wave motion. In no case are the programs intended to replace the traditional labs as they are now.

Hardware: The videocassette tapes were played from a SONY videocassette player onto a SONY Trinitron color television monitor. A DIGI-LOG TeleComputer II keyboard connected with the USNA's time-sharing system displayed the computer output on the SONY monitor. Buttons on the SONY monitor permitted convenient back-and-forth switching between the videocassette and the associated computer program. A "video" button activated the screen for viewing the cassette tape, and a "compute" button put the cassette player in a pause mode while displaying the computer output on the monitor screen.

The monitor had 40 computer characters per line, with a total of 16 lines. The cassette pictures produce good detail and color for showing graphs or close-ups of instruments; and the written dialogue from the computer output is displayed well so that it can be easily read. However, there was not enough resolution in the computer output to provide convenient subscripting for equations from physics and chemistry; and there was not sufficient resolution to use computer-generated graphics on this screen.

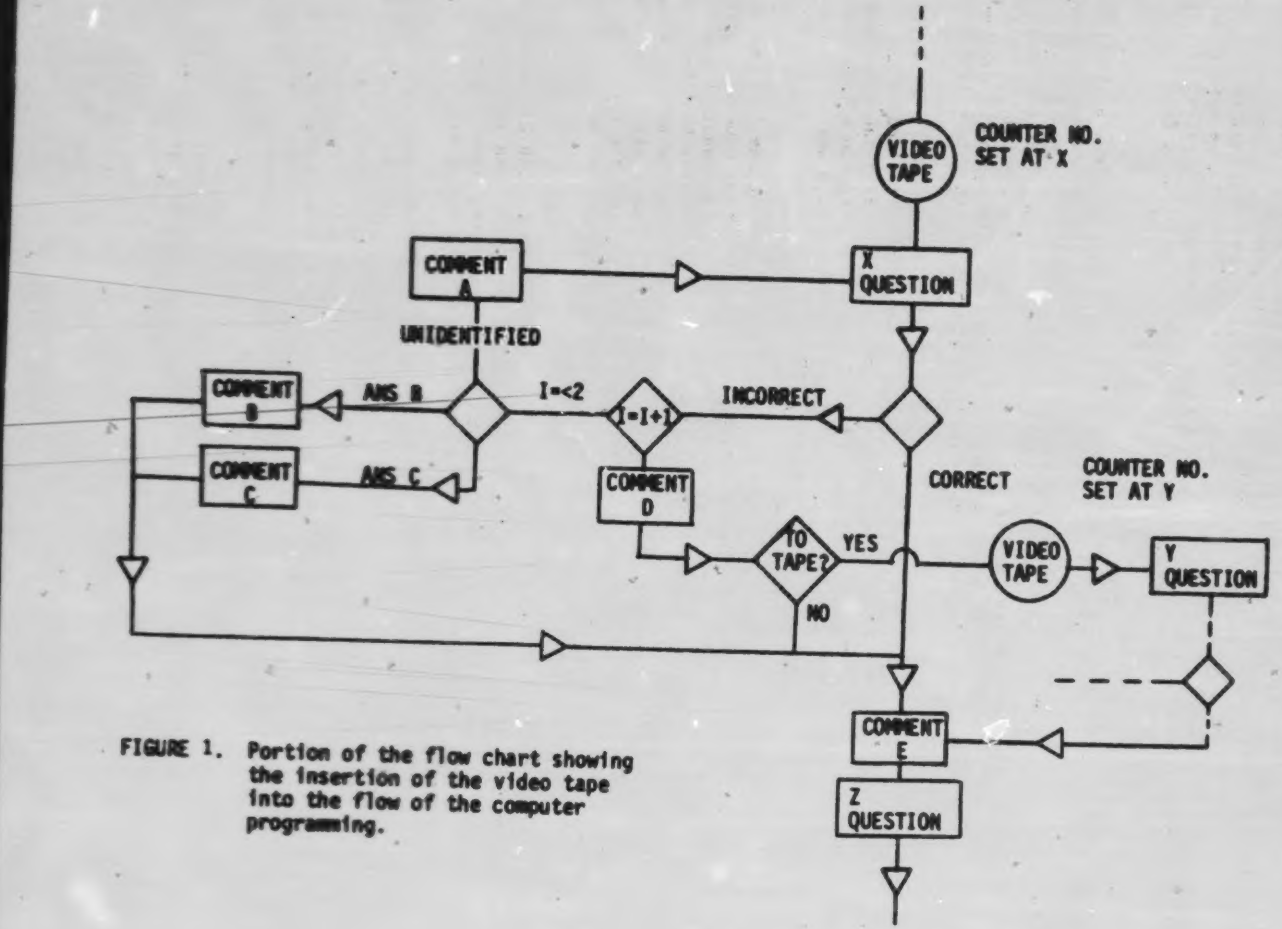

VIDEO TAPE
COUNTER NO. SET AT X
COMMENT A
X QUESTION
UNIDENTIFIED
COMMENT B
ANS B
I=<2
I=I+1
INCORRECT
COMMENT C
ANS C
COMMENT D
CORRECT
COUNTER NO. SET AT Y
TO TAPE?
YES
NO
VIDEO TAPE
Y QUESTION
COMMENT E
Z QUESTION

FIGURE 1. Portion of the flow chart showing the insertion of the video tape into the flow of the computer programming.

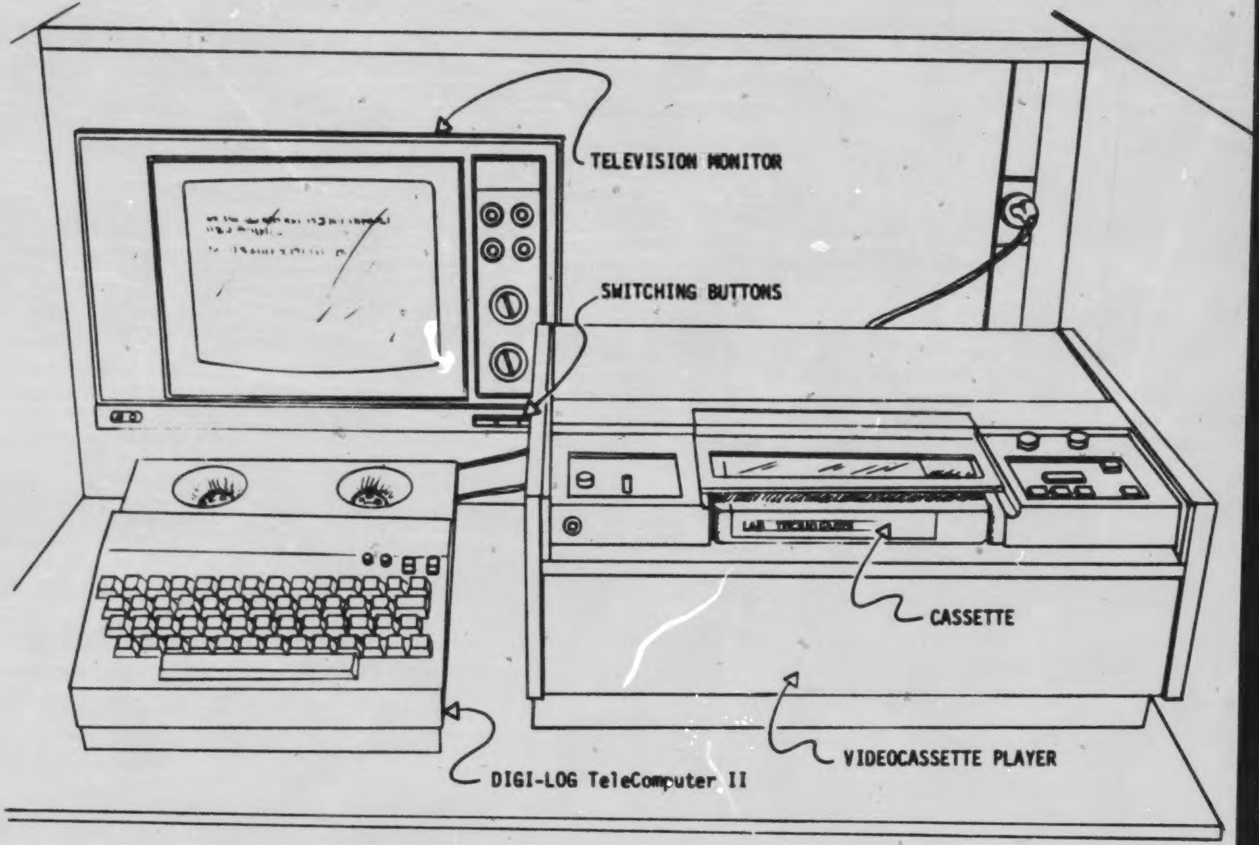

TELEVISION MONITOR
SWITCHING BUTTONS
CASSETTE
VIDEOCASSETTE PLAYER
DIGI-LOG TeleComputer II

FIGURE 2. STUDY CARRELL SHOWING THE ARRANGEMENT OF THE TELEVISION MONITOR, THE VIDEOCASSETTE PLAYER, AND THE COMPUTER KEYBOARD.

A short video tape was prepared which explained the procedure for signing on or for switching between the videocassette and the computer program. Also a much briefer message was placed at the beginning of each video tape just as a reminder in case it was needed. An arrangement of the monitor, videocassette player, and computer keyboard in a study carrel is shown in Figure 2.

Conclusions: Videocassette tapes and time-sharing computer programs can be merged to produce a smooth-flowing, interactive dialogue between them. This combination has the potential to become an effective teaching device for many of the concepts in the basic physics course. Tests are currently being prepared which will give some indication of the effectiveness of these media in combination.

Both the television studio work and the development of the computer dialogue are pretty straightforward, but the preparation is time-consuming for the amount of viewing time produced. Work needs to be done on means of simplifying the total development of these programs so that a number of programs could be tested for use in the physics lab and in the introductory physics course. Programs of problem drills and direct exposition of difficult points of understanding in physics need to be tested. Consideration should be given to different television techniques, such as use of video discs, and to refinement of computer software.

Acknowledgement: The development of these programs was the result of much cooperation between the USNA Educational Resources Center television studio and the Academic Computing Center. This was encouraged and made possible by J. C. Thompson of ERC and by R. A. Pollak of ACC.

COMPUTER SUPPLEMENTED CALCULUS INSTRUCTION

Robert A. McCoy

Southern Oregon State College

INTRODUCTION

Calculus instruction has always made use of diagrams and graphs to illustrate some theorem or concept that the instructor wished to put across to his students. Graphs in texts helped somewhat but these are static, unchanging and provide little, if any, sense of motion. Also to portray another example the instructor would have to produce the graph or graphs for the class.

Prior to 1973 I had used some computer programs to supplement some of the instruction. An example of an early program was the program SERIES 1, which was a very simple computing program to list N, the Nth term of a sequence and the Nth partial sum of the resulting series. By properly selecting examples the instructor could have every student calculate many terms and partial sums and reach conjectures concerning convergence or divergence. Produced at the same time were programs TRAP and SIMP to give trapezoid and Simpson's rule approximations to the definite integral. In the summer of 1973 I participated in a NSF workshop at Carleton College, Northfield, Minnesota under the direction of Dr. Roger Kirchner. The goal of the project was to write programs which would produce graphic displays which would then be filmed (35 mm), packaged and sold as instructional aids. It became apparent to me very early that we were losing much of the action involved, so in the summer of 1974 most of us returned to produce Super 8 film loops. This retained the action but we then had the same example over and over. During those two summers I felt that the dimension needed was the interactive one. Could interactive graphics programs be written to supplement and provide the variety of examples needed in the typical calculus course? I was also faced with making this supplemental material just that, supplemental. The course must not depend on the use of the computer or knowledge of programming. This provided the framework for a proposal to NSF for matching funds to purchase the necessary equipment. Not all of the programs would use graphics so the equipment procured with NSF funds and Tektronics assistance consisted of:

2 -- Dec Writer II LA36
3 -- Tektronics 4012
1 -- Tektronics 4631 Hard Copy Unit
1 -- Tektronics 4953 Graphics Tablet

This project was also enhanced by the fact that Southern Oregon State College acquired a new Harris 200 Computer for installation in May 1976. Much time was spent in acquisition, but in Fall 1976 we were ready to phase in the project. At this writing not all of the programs have been finished and many have been

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

rewritten. They all follow a similar format which aids the student in inserting the required function and running the program.

THE INDIVIDUAL ACTIVITY

Each program was written to illustrate one principle or concept. There was no attempt to write one overall graphics system to do all things for all people. Each program was written to do only one thing and usually planned to be used at a specific point in the instruction. The programs were all written in BASIC using Fortran subroutines in PLOT10 which can be called using the HARRIS BASIC.

The student only needed to know how to log on the computer and write the function definition to be used. The format for this was always the same as illustrated below:

```
USER #
1699M151, JOE
GOOD MORNING JOE COLLEGE
BASIC
OLD, DEVSLOPE
200 DEF FNF(X) = 3*X↑2+2*X+2
CPLOT - A MACRO written to compile the program with appropriate
parts of PLOT10.
```

Everything else required was interactive in response to questions asked by the program.

There were some programs written which were utility type programs. . . examples were COORD, a program which produces coordinates which the student uses to graph a particular equation, and GRAPH, a program which uses the graphics terminal to display the graph. These were intended for use throughout the course and have indeed found use by other departments on the campus.

The rest of the programs were written to illustrate a particular concept. For example, consider DEVSLOPE. This program allows the student to enter the function they wish and then will select a point where they wish to approximate the slope of the tangent to the graph. The program will then graph the function, a series of ten secant lines through the point the student selects and another series of points starting at some distance less than or greater than the selected value and record the slopes of each of the secant lines for student study. The activity sheet and graphics results are on the pages which follow.

PROGRESS TO DATE

First it appears this will be an ongoing project. Hardly a day goes by without a suggestion for a program or program improvement by an instructor or student. Two such programs suggested were RECIPF which graphs a function and its reciprocal on the same screen for comparison and ABSF, a similar program for a function and its absolute value. The programs, particularly GRAPH, have found their way into Physics, Chemistry and Economics courses.

As a supplement to the Calculus the project seems to be useful. Some of my colleagues have used the activities extensively while others have used them less frequently. As this year progresses, all Calculus courses and the pre-Calculus functions course will be to some degree supplemented with this project. As for student reaction, so far it is mixed. There are those few students to whom the thought of sitting down at a computer terminal was frightening at first but the fears were soon dispelled. The other extreme are the students who wish to spend all of their time at the terminal. However, the majority of the students find the computer assistance useful. One major example involved the use of DEVPLTG prior to a study of critical points, MAX-MIN, points of inflection, etc., to arrive at student conjectures concerning these concepts. The students then had much more interest in proving THEIR conjectures. Each activity sheet outlines the objective for that exercise but more often than not some students arrive at some conjectures very early. For example, while using DEVCOMP, a first term student came to my office to report he had concluded that DxSIN(X) was COS(X) and he was ready to find proof then and there. I was happy to oblige him. There are some problems with the use of graphics but these can also be put to good use if approached properly. Example: In graphing SIN(X)/X, more likely than not, the computer will graph directly through the discontinuity at X = 0. By explaining the method used for graphing from point to point with short line segments you can show that the graph produced can indeed lie to you at times creating a need for the student to be aware of such discontinuities themselves.

Another conclusion I, as one instructor, have arrived at is that the activity sheets need to be more flexible for use by other instructors. Examples I use may not be the examples another instructor might select. Therefore, I am presently reworking the activity sheets to allow for each instructor to adapt the individual program to his or her use.

SUMMARY

It is not my intention to advocate that everyone change to using computers to assist them in their Calculus instruction but rather to make them aware of my attempt to supplement our Calculus instruction here at Southern Oregon State College and perhaps to inspire others to try a similar approach with the goal of developing some useful computer graphics tools.

COMPUTER LAB ACTIVITY NUMBER 7 CALCULUS I
TITLE Introduction to Derivatives
OBJECTIVE: To enable the student to gain an intuitive feeling for the definition of the derivative.
COMPUTER ACTIVITY:

Log on
BASIC
OLD, DEVSLOPE
200 DEF FNF(X) = X + 3 or any function you want.
CPLOT

(This is a graphic program and must be run on a graphics terminal) This program will graph ten (10) secant lines from (x,f(x)) to (x+h, f(x+h)) with h reduced by one-half each time, and will compute the slope of each secant line. Watch for these values to approach some limit. You may take h either positive or negative.

You may repeat and get better approximations by taking h smaller.

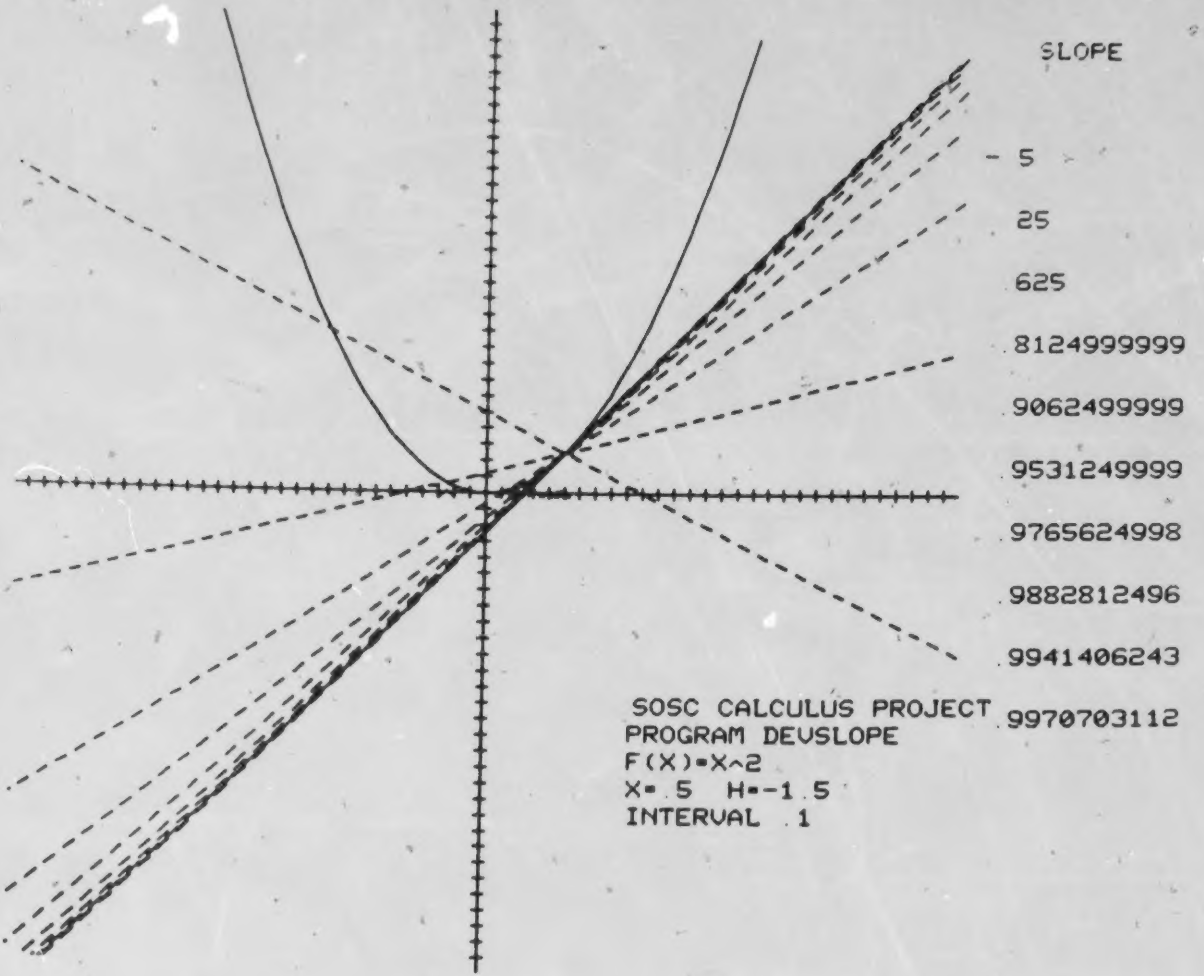
Repeat for all functions assigned.

Log Off

Functions to use:
y = X + 2 at .5 Distance +1.5 and -1.5
y = SQR(X-2) at 2 Distance +2
Others assigned by your instructor.

CONCLUSIONS:

1. Did it make any difference with y = X + 2 with distance set at 1.5 or - 1.5?
2. What did slopes approach in y - SQR (X-2)?



```
-.5
.25
.625
.8124999999
.9062499999
.9531249999
.9765624998
.9882812496
.9941406243
.9970703112
```

```
SOSC CALCULUS PROJECT
PROGRAM DEVSLOPE
F(X)=X^2
X= .5  H=-1.5
INTERVAL .1
```

MATHDOC: A CONTROL SYSTEM FOR COMPUTER ASSISTED CALCULUS

Mark J. Christensen

Georgia Institute of Technology

1. General Description

MATHDOC is a control and documentation system used by the School of Mathematics of the Georgia Institute of Technology in conjunction with its calculus demonstration programs. In particular MATHDOC is used by the student to obtain documentation (complete with text references) about the operation and usage of the calculus demonstration programs. Additionally the student may run these programs from a terminal (of which there are approximately 160 on this campus) while under the MATHDOC system. Thus MATHDOC acts as an intermediate device between the user and the operating system of our CDC CYBER 74.

This is an improvement over the system used at this Institute in the past. The system previously used here, as at many other institutions, involved the distribution of duplicated material on each program, often supplemented by lectures. The disadvantages of these methods of distribution of information are great both economically and pedegagically. One of the primary advantages of MATHDOC is that a single printed sheet (see Appendix I) is distributed at the start of the term. Thereafter the only investment of class time that is required is that necessary for the instructor to state the problem in the most general terms. It is then left for the student, with the aid of MATHDOC, to select a program suitable for the solution of the problem and obtain the information necessary for its operation. For this reason MATHDOC contains a catalog of the programs currently available. The entire process of program selection, investigation of the documentation and operation of the actual program is carried out under MATHDOC and is accomplished by the use of the commands 'YES', 'NO' and 'programname'. Thus the command structure of the computer itself is eliminated as much as possible. Indeed, this was one of the primary goals of our design efforts. A sample run of MATHDOC exhibiting these and other features will be presented in the following section.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

Ⓒ *1978 CCUC*

The MATHDOC system, as presently constituted, was developed by the author of this article and an undergraduate programming aide, Mr. Donald Lutz, during the year 1975-76. The original demonstration programs were provided by a former staff member Dr. Samuel Bernstein during the year 1974-75. In order to take advantage of the features of MATHDOC it was necessary to rewrite many of these programs in the course of the last year, as well as introduce several new ones.

These programs by and large address themselves to the needs fo the first two quarters of the calculus sequence.

2. The Setting and Operation.

At this Institute approximately 2,000 freshmen are admitted each year. Of these, approximately 90% take the calculus sequence during their first and second year. Therefore, the number of potential users for the system is very large. Additionally, since most of the students will become engineers or scientists, an early and relatively painless introduction to computers is desirable. Usage statistics compiled during the first quarter of the academic year 1976-77 show student usage when MATHDOC became available. (See Table 1, Appendix II). It should be noted however that the average per student interactive time per week (that is, the average amount of time each student was actually connected to the computer) is of the order of one hour/week/student.

Thus the MATHDOC system seems to promote the use of the computer as a tool for the illustration of the principles of the calculus and the development of examples for the student. One of the most striking ways in which this is accomplished is by means of the documentation feature of MATHDOC. That is to say, if the student wishes to know how a program operates without examining the source code (which he may or may not know how to interpret), he merely requests information on that program from MATHDOC. Additionally this will provide the student with an explanation of the questions he will be asked by the program, as well as a sample run of the program. As an example of these features we include a copy of a sample run of a MATHDOC session in Appendix V. The responses of the user are underlined.

This example shows several of the features of the MATHDOC system. First, the only commands that the student issues to the computer operating system are "GET,MATHDOC/UN=MATHDAA" and "-MATHDOC". Secondly, when the student requests information on the program BERNSON he receives only one paragraph of information and is then asked if he desires more. This cycle is repeated several times. The idea with this feature is to avoid unnecessary printing at the terminal. We remark that if the student had responded with "YES" when asked "DO YOU WANT MORE ON BERNSON?" instead of "NO" on page ii, line 32 he would have received a copy of a sample run of this program. The actual run of BERNSON itself is self-explanatory, with two exceptions: If the student had answered 'NO' when asked 'DO YOU KNOW HOW TO WRITE A FUNCTION IN BASIC?' he would have been instructed to request information on BASIC. This function operates in the same way as the documentation feature for the programs. Secondly, when the student typed '100 DEF FNA....' this string was written on a short file which was then APPENDED and PACKED with the local copy of the actual program file. MATHDOC then submits this combined file to CYBER. This feature, along with several others, is described in the diagrams in Appendix IV.

In the other calculus demonstration systems available on this campus and the University System of Georgia all documentation is in the form of manuals or duplicated sheets. We feel that the self-documentation feature of MATHDOC, along with the function load feature, is one of its primary assets.

Also, the above example shows, our goal is to have the demonstration programs display as many of the intermediate steps as possilbe instead of merely displaying a number. That is to say, rather than have the program accept the students' data, compute a final result based on that data and then print it we have attempted to present the student with as many of the intermediate steps as possible. Our goal here has been to make a session with the computer as much like a lecture as possible so that the student can go over his output later to increase his understanding of what has happened. To achieve this level of documentation and program excellence requires that great care be given to each phase of program construction. Thus the writing of even a simple program can involve several days of careful study and composition, as well as a series of exhaustive trials.

3. Concerning the Appendices.

Appendix I is a copy of the handout given to each student at the time his/her computer ID. Experience has shown that this single handout is sufficient.

Appendix II consists of two brief tables of usage statistics.

Appendix III is a copy of the MATHDOC catalog.

The three figures in Appendix IV illustrate some of the system functions of MATHDOC. In each figure the key unit is the program QUERY. QUERY communicates with the user and determines which system function is to be implemented. In Figure 1, for example, the user has requested information on a specific program. In this case QUERY obtains the data file for this program and passes it to DEMOPRT which outputs the data to the user. In Figures 2 and 3 the situation is more complicated. If a program is to be executed QUERY writes a file (PROC) and passes the program file (X) to it for execution. If the program needs a function from the user, as is the case in Figure 3, then PROC itself writes a brief file (FUNC') and appends it to the basic program file. Once this has been done the combined file is submitted to the CYBER for execution. In either case, once the program is finished, control is returned to QUERY which then purges PROC and the local copy of the program.

Finally it should be stated that the MATHDOC system is written in BASIC and uses the string manipulation features of that language extensively. Certain functions use the NOS control language.

Appendix I

MATHDOC

G.T. SCHOOL OF MATHEMATICS

CALCULUS DEMONSTRATION SYSTEM

MATHDOC is a system intended to allow the student to operate the calculus demonstration programs with a minimum of knowledge of the computer. Once you have obtained your computer user number and password from your instructor you should go to a terminal location and turn on one of the terminals. The method for doing this varies, so be sure and have someone at the terminal site you have selected show you how to do so. Once the terminal is operating you will enter your user number and password when so instructed by the computer.

The following is a sample run with MATHDOC. The responses of the user are here typed in lower case. Comments are in brackets.

```
76/10/12.  09.04.06.

GEORGIA TECH CYBER 74                         NOS 1.1-420/423

USER NUMBER: mannnaa  ( type in your user number here )

PASSWORD          ( type in your password over the mask )

TERMINAL:     76, TTY

RECOVER/SYSTEM:  get,mathdoc/un=mathdaa


READY

-mathdoc


MATHDOC MARK II.1  09/12/12.  09.06.45

(from here on your with MATHDOC)-------------------
(when you wish to stop, just type QUIT)


THANKS FOR CALLING US.  GOOD LUCK....
SRU   5.789 UNITS
bye   ( typing bye, when out of mathdoc, disconnects you from the
           computer )
```

In order to send messages to the computer after they have been typed hit the return key (new line key on a CRT). If you make an error and realize it before hitting return, hit the esc (escape) key and then retype.

Appendix II

Table 1

Accesses to the MATHDOC system

Period	9/20-10/19/76
Number of users	162
Number of acceses	1589
Accesses/user/day	1.09

NOTE: Three sections of first quarter Caluclus were using the system.

Table 2

Accesses to Program Bernson

Period	9/20-10/17/76
Number of users	104
Accesses/user	2.10

NOTE: More statistics will be presented.

Appendix III

```
DO YOU WISH A CATALOG AND GENERAL INFORMATION ? YES
 09.11.22.      76/10/04.
DOCUMENTATION ON CATALOG

   MATHDOC PROVIDES DOCUMENTATION FOR THE CALCULUS DEMO
PACKAGE.  THE FOLLOWING PROGRAMS ARE AVAILABLE:
PROGRAM NAME          PROGRAM FUNCTION

  FATE                ILLUSTRATES ROUND-OFF ERROR
  PLOTTV              PLOTS A USER SPECIFIED FUNCTION ON CRT SCREEN
  BERNSAU             SUMMING OF SERIES
  BERNSIX             GRAPHING AID FOR RATIONAL FUNCTIONS
  BERNSON             ESTIMATES THE VALUE OF THE DERIVATIVE
  BERNSUM             BRIEF PROGRAM FOR AREA
  BERNSAB             ESTIMATES SLOPE FOR A POLYNOMIAL
  BERNSEG             ILLUSTRATES SIMPSON'S RULE FOR AREAS
  BERNSID             ESTIMATES INTEGRAL BY UPPER SUMS,
                      LOWER SUMS, TRAP. RULE AND SIMPSON'S
  PLOT                PLOTS A USER SPECIFIED FUNCTION AT TELETYPE
  ZERO                ESTIMATES THE ROOTS OF A FUNCTION

DO YOU WANT MORE ON CATALOG ? YES
THE CALLING SEQUENCE FOR ANY OF THESE IS ( WITHOUT MATHDOC ):
BASIC,OLD,PROGRAM NAME/UN=MATHDAA,ND
FOR EXAMPLE, TO GET BERNSAB THE SEQUENCE IS
BASIC,OLD,BERNSAB/UN-MATHDAA,ND
ONCE YOU HAVE OBTAINED A COPY OF THE PROGRAM, YOU NEED
ONLY TYPE THE WORD 'RUN'.
           WHEN IN THE MATHDOC SYSTEM, SIMPLY TELL THE COMPUTER YOU
WISH TO RUN A PROGRAM (WHEN ASKED!) AND THEN ANSWER WITH THE
PROGRAM NAME WHEN THE COMPUTER ASKS WHAT PROGRAM YOU TO RUN.
```

Appendix IV

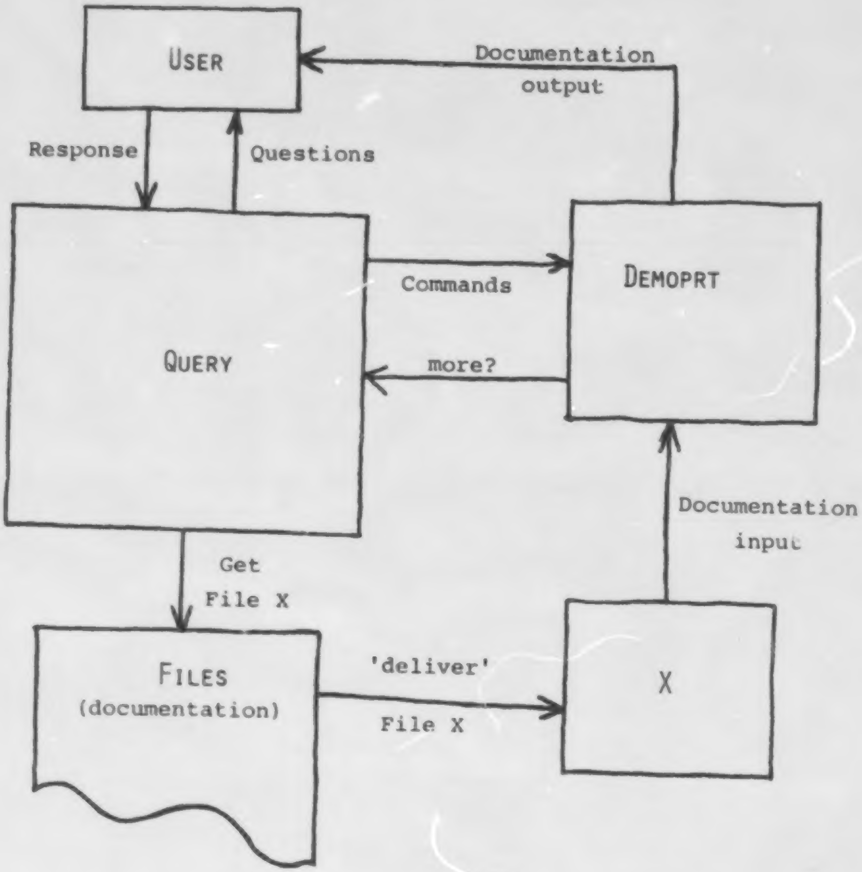

USER
Documentation output
Response
Questions
QUERY
Commands
more?
DEMOPRT
Documentation input
Get File X
FILES (documentation)
'deliver' File X
X

Figure 1 : The Documentation Cycle

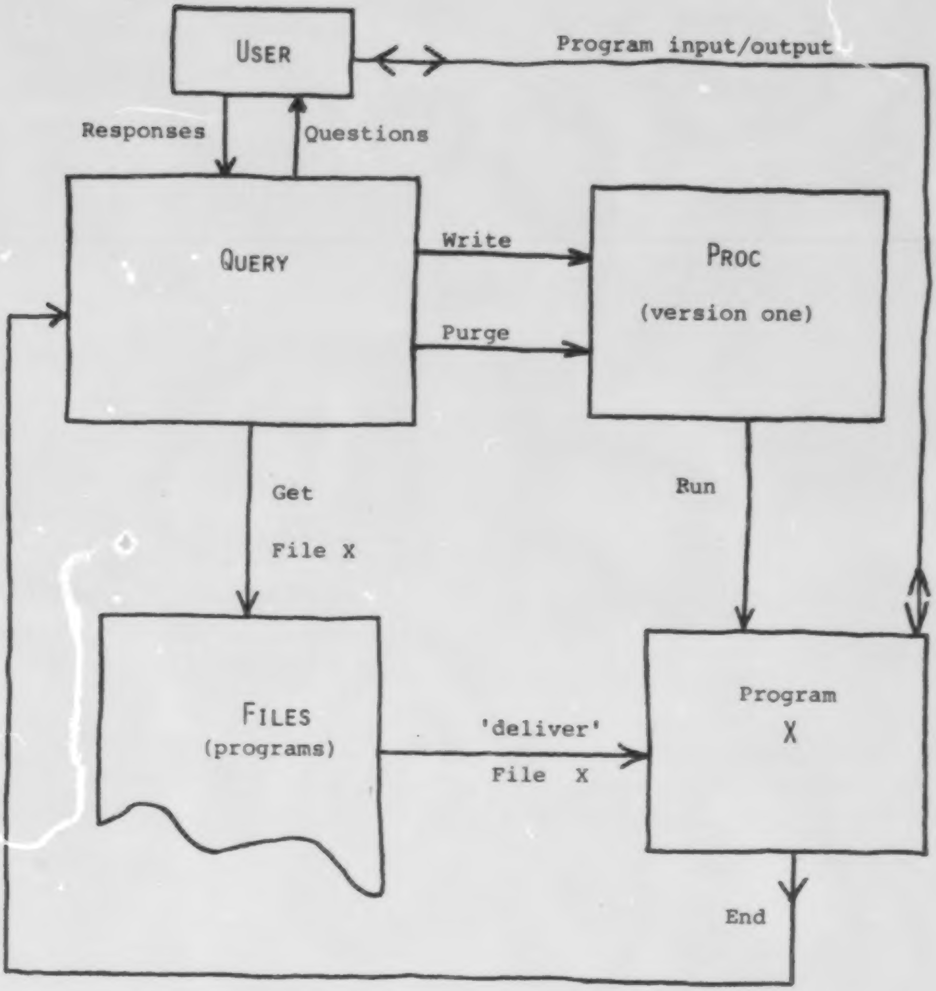
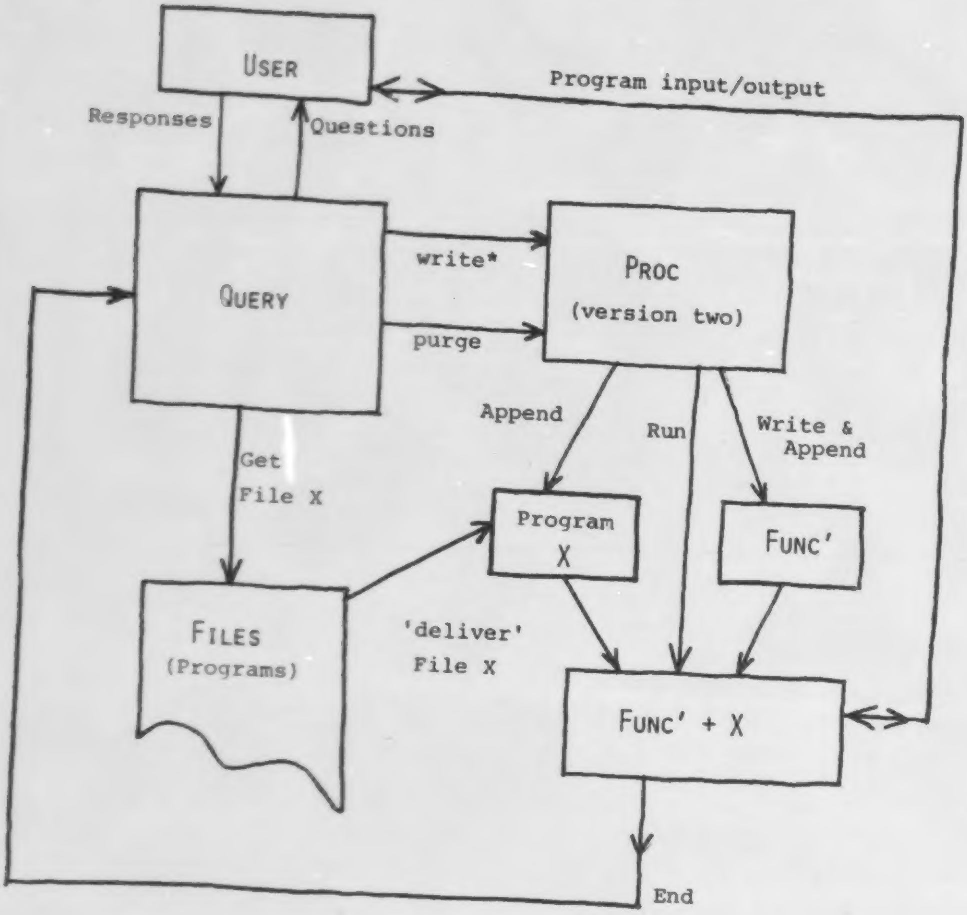

USER
Program input/output
Responses
Questions
QUERY
Write
Purge
PROC
(version one)
Get
File X
Run
FILES
(programs)
'deliver'
File X
Program
X
End

Figure 2 : Program Execution
(w/o function load)


USER
Program input/output
Responses
Questions
QUERY
write*
purge
PROC
(version two)
Append
Run
Write &
Append
Get
File X
Program
X
FUNC'
FILES
(Programs)
'deliver'
File X
FUNC' + X
End

* Including function.

Figure 3 : Program Execution
(with function load)

```
    76/10/29.  11.10.45
    GEORGIA TECH CYBER 74.                    NOS 1.1-430/428.
 USER NUMBER: MA201MC,
 TERMINAL:      74, TTY
 RECOVER /SYSTEM: GET,MATHDOC/UN=MATHDAA


 READY.
 -MATHDOC



 MATHDOC MARK II.1  05/23/76  11.11.59.

    G T SCHOOL OF MATHEMATICS
 CALCULUS DEMONSTRATION SYSTEM


 WOULD YOU LIKE INFORMATION ON MATHDOC?
 (IF THIS IS YOUR FIRST TIME USING MATHDOC, TYPE 'YES')
 ? NO
 DO YOU WISH A CATALOG AND GENERAL INFORMATION ? NO*
 DO YOU WISH INFORMATION ON A SPECIFIC PROGRAM? YES
 WHAT PROGRAM ARE YOU INTERESTED IN? BERNSON

  11.13.07.       76/10/29.
 DOCUMENTATION ON BERNSON

    THE PROGRAM BERNSON ESTIMATES THE DERIVATIVE OF ANY FUNCTION
 (WRITTEN IN BASIC) FOR A VALUE OF X AND H SPECIFIED BY THE USER
 (SEE BONIC, P. 52 AND FIGURE 5)   .  TO RUN THIS PROGRAM WHILE IN
 THE MATHDOC SYSTEM ANSWER 'YES' TO THE QUESTION

           'DO YOU WANT TO RUN A PROGRAM?'

 THEN ANSWER WITH  'BERNSON' WHEN YOU ARE ASKED:

        'WHAT PROGRAM ARE YOU INTERESTED IN?'

    YOU WILL THEN BE ASKED TO SUPPLY THE FUNCTION WITH THE LINE

    ' 100 DEF FNA(X)=----THE FUNCTION IN BASIC-----

 DO YOU WANT MORE ON BERNSON ? YES
```

---

* A copy of the current catalog is given in Appendix III

```
     TO RUN BERNSON WITHOUT MATHDOC YOU SHOULD, AFTER THE
COMPUTER TYPES 'READY' RESPOND WITH:

          'BASIC,OLD,BERNSON/UN=MATHDAA'

     AFTER THE COMPUTER TYPES 'READY', ENTER THE FUNCTION AS
PREVIOUSLY EXPLAINED. THEN TYPE 'RUN'.

DO YOU WANT MORE ON BERNSON ? YES


     BESIDES THE FUNCTION YOU MUST SUPPLY (AFTER THE RUN
HAS BEGUN) THE VALUES OF X AND H.  THE COMPUTER WILL THEN
CALCULATE THE DIFFERENCE (OR NEWTON) QUOTIENT AND PRINT IT
OUT AT THE TERMINAL.  NEXT YOU WILL BE ASKED :

        'DO YOU WISH TO CHAGE H?'

TO THIS QUESTION YOU MAY ANSWER WITH

1.) 'YES' AND THEN CHANGE H WHEN ASKED.
2.) 'NO' . IN THIS CASE THE COMPUTER WILL THEN ASK 'DO YOU WISH
     TO CHANGE X?'  THIS MAY, IN TURN, BE ANSWERED WITH YES OR NO.
3.) 'AUTO'  .  THIS RESPONSE WILL CAUSE THE COMPUTER TO
    DIVIDE YOUR CURRENT VALUE OF H BY 2 AND REPEAT THE COMPUTATION
    OF THE NEWTON QUOTIENT FOR THIS NEW VALUE OF H.  THIS
    PROCESS WILL BE REPEATED 5 TIMES.  AFTER THIS HAS BEEN
    DONE YOU WILL AGAIN BE ASKED 'DO YOU WISH TO CHANGE H?'

DO YOU WANT MORE ON BERNSON ? NO


DO YOU WISH TO RUN A PROGRAM? YES
WHAT PROGRAM WOULD YOU LIKE TO RUN? BERNSON
DO YOU KNOW HOW TO WRITE A FUNCTION IN BASIC? YES
INPUT FUNCTION DEFINITION (IN BASIC)
(REMEMBER TO START WITH '100 DEF FNA(X)= ')
? 100 DEF FNA(X)=X*COS(X**2)

 76/10/29.  11.19.29.
PROGRAM     BERNSON
```

```
THIS PROGRAM WILL COMPUTE THE DIFFERENCE QUOTIENT
AND HENCE ESTIMATE THE DERIVATIVE FOR THE FUNCTION

          F(X) = X*COS(X**2)

FOR VALUES OF X AND H THAT YOU WILL SPECIFY.

FOR WHAT VALUE OF X DO YOU WISH TO ESTIMATE THE DERIVATIVE
? 0

FOR X = 0  WE HAVE F( 0 ) = 0

WHAT VALUE OF H DO YOU WISH TO USE?  .03

THE ESTIMATE OF F'( 0 ) FOR H = .03  IS

(f( 0 + .03 ) - F( 0 ))/ .03  =

( .03 - ( 0 ))/ .03  =  1.

DO YOU WISH TO CHANGE H ? NO
DO YOU WISH TO CHANGE X ? YES

FOR WHAT VALUE OF X DO YOU WISH TO ESTIMATE THE DERIVATIVE
? 4

FOR X = 4 WE HAVE F( 4 ) = -3.83064

WHAT VALUE OF H DO YOU WISH TO USE? .05

THE ESTIMATE OF F'( 4 ) FOR H = .05  IS

(F( 4 + .05 ) - F( 4 ))/ .05 =

(-3.11182 - (-3.83064 ))/ .05 = 14. 3764

DO YOU WISH TO CHANGE H ? AUTO


THE ESTIMATE OF F'( 4 ) FOR H = .025  IS

(F( 4 + .025 ) - F( 4 ))/ .025 =
(-3.54634 - (-3.83064 ))/ .025 = 11.3721
        (the computation is repeated five times)
DO YOU WISH TO CHANGE H ? NO
DO YOU WISH TO CHANGE X ? NO

THIS COMPLETES THE ESTIMATION OF THE DERIVATIVE OF
X*COS(X**2)  AT   4  .
```

END OF EXAMPLE

TIGERS AND RABBITS:
CURRICULAR USES OF A SYSTEM DYNAMICS
MODEL OF A SIMPLE IDEALIZED ECOSYSTEM

PRISCILLA W. LAWS

DEPARTMENT OF PHYSICS AND ASTRONOMY

DICKINSON COLLEGE

CARLISLE, PA. 17013

I. INTRODUCTION

The Computer model described in this paper was originally developed in 1972 to enable students in an interdisciplinary course, entitled "environment, culture and values"[1] to understand more fully the methods and principle conclusions of the World 3 global model developed at MIT for the Club of Rome's project on the predicament of mankind.

In the past 6 years it has been used by the author's colleagues in a number of other undergraduate courses at Dickinson including Environmental Science, Game Theory and Mathematical Modeling. Several students doing independent studies in System Dynamics Modeling and Environmental Systems have also used the model. (These uses are described at the conclusion of this paper.)

The method used to construct this model of a simplified island ecosystem is based entirely on the method of system dynamics developed by Forrester.[2,3] This method was used extensively by a research team at MIT[4,5] to study the possible future behavior of the global system interrelating factors such as population, food, natural resources, capital investment, and pollution. (A simplified instructional program has also been written in BASIC to simulate the global system as part of the Huntington II Project.[6])

The actual behavior of the ecosystem described in this paper is similar to that of a number of currently available computer models of preditor-prey systems.[7,8] However, its basic structure and output format are similar to those used by Forrester and Meadows, et. al., in their global models.

The model is programmed in FORTRAN IV and has been run on 3 different Computing Systems at Dickinson College - the IBM 1130, the Interdata 7/32, and the DEC PDP - 11/55. It is currently being run on the DEC 11/55 RSTS/E timesharing

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

system. The program is available by request on 9 track 800 b.p.i. magnetic tape. A nominal fee to cover costs is charged.

The program enables individuals with no previous computer experience to explore the behavior of an idealized ecosystem over time under different conditions. The user is allowed to make simple changes in the initial populations of tigers and rabbits and/or their birth and death rates.

II. DESCRIPTION OF THE ECOSYSTEM

A hypothetical tropical island produces a fixed amount of green plants each month. These plants are eaten by rabbits, which populate the island. However, any plant matter which is not eaten by the rabbits in a given month decomposes. Rabbits are eaten by tigers. A diagram of the ecocycle is shown in Figure 1. Each tiger stakes out his own territory and hunts for rabbits. The probability that a tiger will see and then catch a rabbit is proportional to the density of rabbits in his territory at a given time.

Under "normal" circumstances the death and birth rates for tigers and rabbits remain constant and the initial populations are set at 2000 rabbits and 40 tigers. At this rabbit density each tiger is able to catch and eat 25 rabbits in a month or half the rabbits in his territory.

III. DESCRIPTION OF THE MODEL

The goal of the system dynamics model is to describe the relationship between the tiger population and the rabbit population and see how these population levels change in time. For example, if the tiger population is known for a given month then its population the next month is found by adding the number of tigers born and subtracting the number of tigers that die. If the birth and death rates are known for the tigers under normal conditions (2000 rabbits and 40 tigers) then the new tiger population can be computed. If, however, there are more (fewer) tigers than normal then one would expect the birth rate of tigers to decrease below normal and the death rate to increase (above normal) because the tigers are not as well nourished. The model is used to describe in detail how the birth and death rates of tigers change under certain circumstances by allowing for the computation of multipliers which change the birth and death rates under different conditions.

Similarly, the rabbit population increases with the birth of new rabbits and decreases with the death of old rabbits either by natural means (disease, old age, etc.) or because they are consumed by tigers. Again deviations from normal conditions on the island change the birth and death rates of rabbits in a predictable way by allowing for the computations of multipliers which change the birth and death rates.

The symbols used in a System Dynamics Flow Diagram are displayed in Figure 2. A complete flow diagram of the model is shown in Figure 3.

IV. THE STRUCTURE OF THE MODEL

1. Tiger Population TP (P on output graph)

The tiger population at the beginning of any month is calculated as the population at the beginning of the previous month, plus the tigers added by the birth rate during the previous month, minus the tigers removed by the death rate.

The tiger population is a system level which is needed to compute the tiger food ratio (TFR) and the tiger quality of life (TQL).

$$TP(K) = TP(I) + DT * [TBR(I) - TDR(I)]$$

Where:

TP(K) - tiger population for month K = I + 1
TP(I) - tiger population for month I
DT - time interval = one month

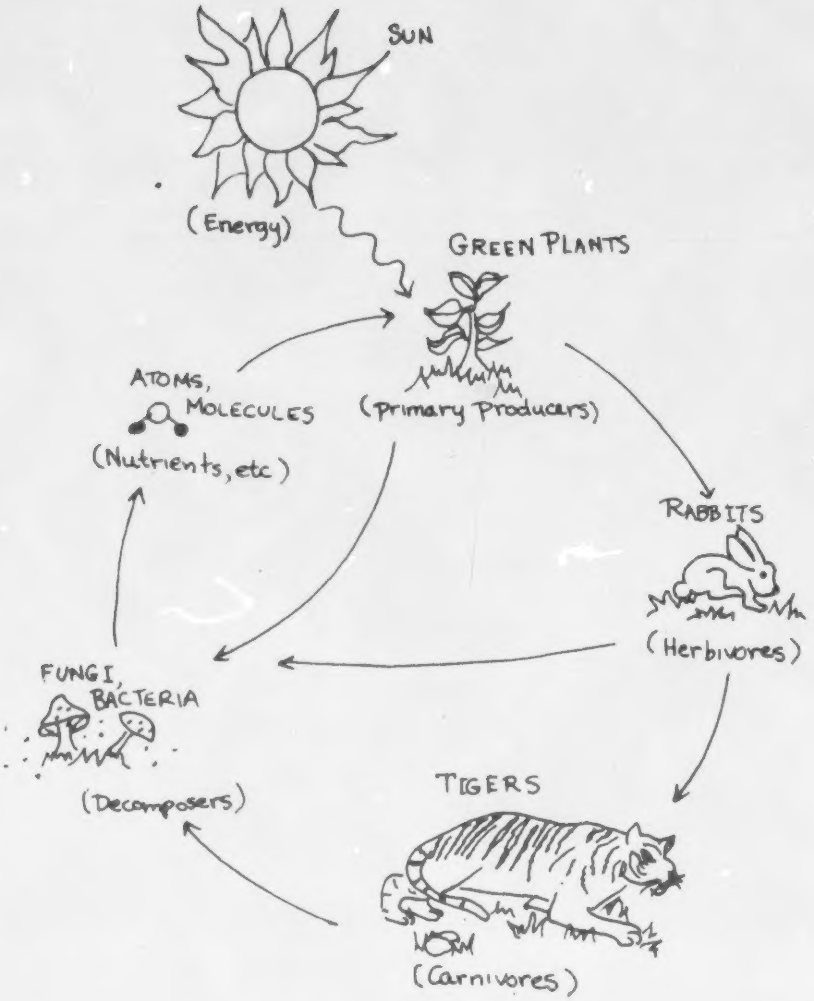

SUN
(Energy)
GREEN PLANTS
(Primary Producers)
ATOMS,
MOLECULES
(Nutrients, etc)
RABBITS
(Herbivores)
FUNGI,
BACTERIA
(Decomposers)
TIGERS
(Carnivores)

Figure 1: A diagram of the terrestrial ecosystem describing the coexistence of tigers, rabbits, green plants, and decomposers on an imaginary tropical island.

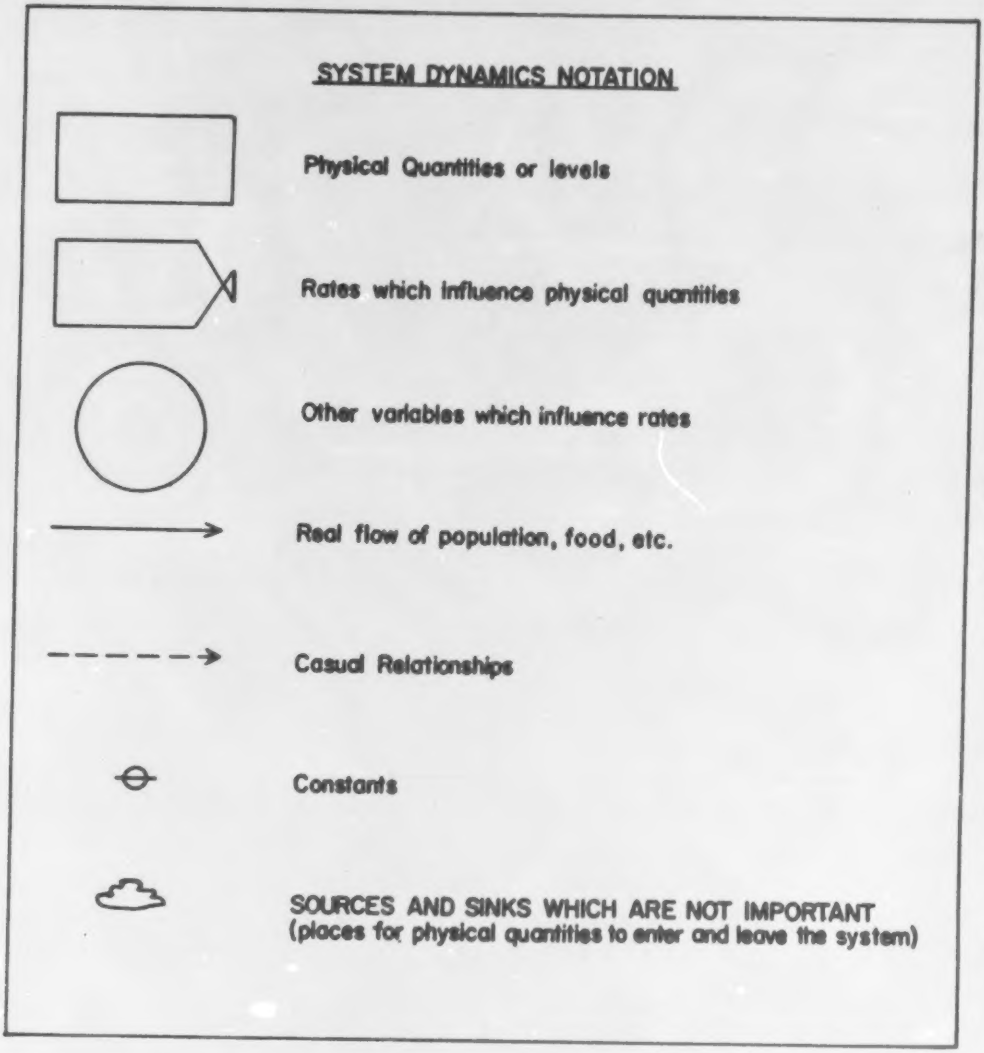

SYSTEM DYNAMICS NOTATION
Physical Quantities or levels
Rates which influence physical quantities
Other variables which influence rates
Real flow of population, food, etc.
Casual Relationships
Constants
SOURCES AND SINKS WHICH ARE NOT IMPORTANT
(places for physical quantities to enter and leave the system)

Figure 2: System Dynamics notation for flow diagrams.

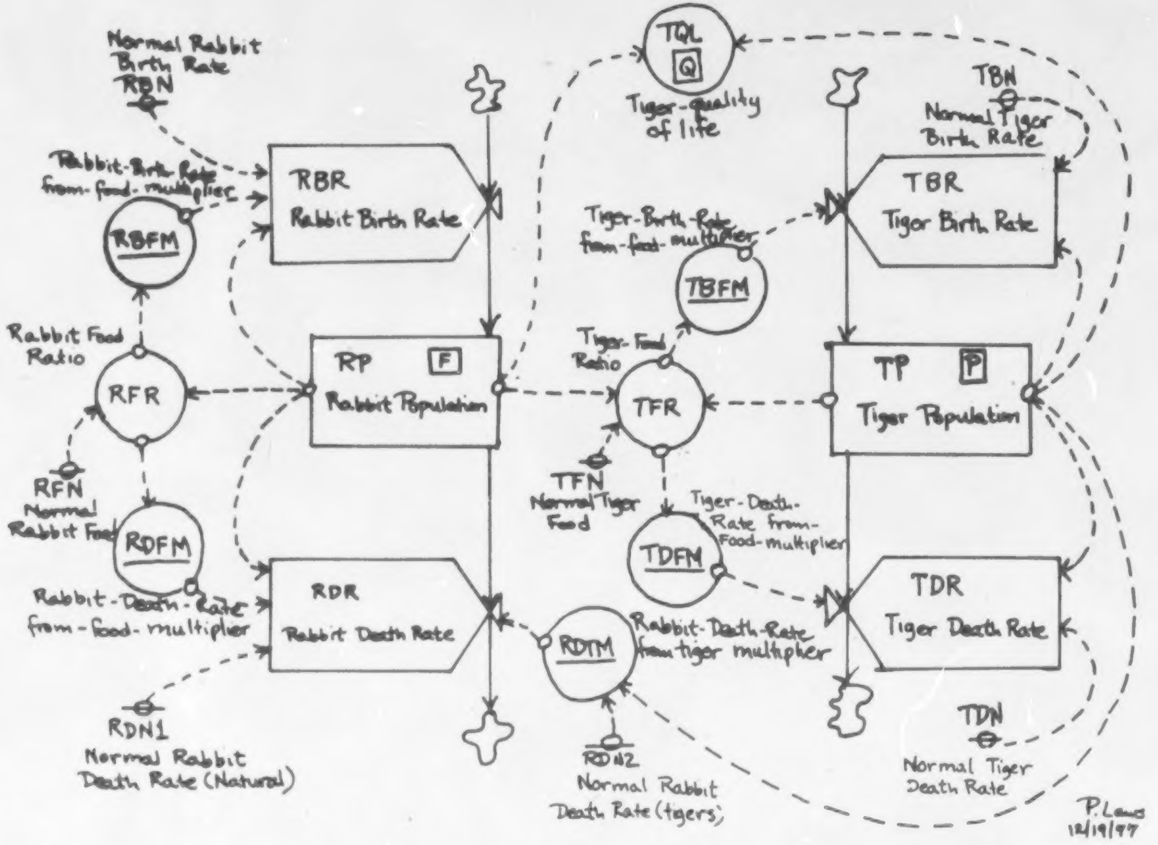

Normal Rabbit Birth Rate
RBN
Rabbit-Birth-Rate from-food-multiplier
RBFM
RBR
Rabbit Birth Rate
Rabbit Food Ratio
RFR
RFN
Normal Rabbit Food
RDFM
Rabbit-Death-Rate from-food-multiplier
RDR
Rabbit Death Rate
RDN1
Normal Rabbit Death Rate (Natural)
RP
F
Rabbit Population
RDTM
Rabbit-Death-Rate from tiger multiplier
RDN2
Normal Rabbit Death Rate (tigers)
TQL
Q
Tiger-quality of life
Tiger-Birth-Rate from-food-multiplier
TBFM
Tiger-Food Ratio
TFR
TFN
Normal Tiger Food
Tiger-Death-Rate from-Food-multiplier
TDFM
TBN
Normal Tiger Birth Rate
TBR
Tiger Birth Rate
TP
P
Tiger Population
TDR
Tiger Death Rate
TDN
Normal Tiger Death Rate
P. Laws
12/19/77

Figure 3: Flow Diagram of the model interrelating the two level variables - tiger population and rabbit population.

TBR(I) - tiger birth rate during the I^{th} month
TDR(I) - tiger death rate during the I^{th} month

2. Tiger Birth Rate TBR

The birth rate is part of the positive-feedback loop shown in the flow diagram. Under normal conditions the birth rate depends on the product of tiger population (TP) and the normal birth rate (TBN). If conditions and hence tiger food are different than basic birth rate changes according to the tiger-birth-rate-from-food multiplier (TBFM) which is explained in the next section.

$$TBR(I) = TP(I) * TBN \times TBFM$$

Where:

TBN - normal tiger birth rate (no. of tigers born/month) when there are 2000 rabbits and 40 tigers so that each tiger eats 25 rabbits/mo.
TBFM - tiger birth-rate-from-food multiplier

The time delay in birth rate changes due to the gestation period for tigers is neglected. The human gestation period is less than a year. By assuming a hypothetical tiger gestation period of less than a month, monthly changes in tiger population could still be compared to the yearly changes in the human population described in the World 2 and World 3 global models.

3. Tiger-food-ratio TFR

The tiger-birth-ratio-from-food multiplier depends on the proportion of available food, and a tiger food ratio can be defined as a dimensionless quantity which compares the amount of food eaten each month per tiger to the amount eaten under "normal" circumstances. Under normal conditions the island holds 2000 rabbits and 40 tigers so that each tiger is alloted 50 rabbits. Under these conditions a tiger is only able to capture half his available rabbits i.e., 25 rabbits/mo each month, since his hunting techniques are not perfect. Thus a TFR of 1 means 25 rabbits were consumed while a TFR of 2 means twice as many or 50 rabbits were consumed.

In attempting to make reasonable assumptions about how tigers hunt the tiger-food-ratio did not turn out to be proportional to a simple ratio of the tiger and rabbit populations. It was assumed that the probability of a tiger capturing a rabbit at a given moment was proportional to the density of rabbits in his "territory".

Thus an island populated with 1 tiger and 10 rabbits would not be as easy for tigers to hunt on as an island populated with 10 tigers and 100 rabbits because the lone tiger would have to cover more ground to capture his rabbits.

Assume that the number of rabbits captured is proportional to the density of rabbits in a tiger territory and this density decreases during the month as the rabbits are "picked off" (new rabbits born during the month do not emerge from their holes until the beginning of the next month). The "decay" of rabbits is exponential (similar to radioactive "decay") and occurs more rapidly as the number of tigers increases.

The number of rabbits captured by a tiger in the I^{th} month (DN) turns out, under the above assumption, to be equal to

$$DN = \frac{\text{rabbit pop in } I^{th} \text{ month}}{\text{tiger pop in } I^{th} \text{ month}} \left(1 - \exp\left[(\text{tiger pop})\ (1\text{mo}) \times \frac{\ln 2}{40}\right]\right)$$

Then the tiger food ratio (TFR) is the ratio of the actual number captured in the I^{th} month (DN) to the number which would have been captured under normal conditions (25 rabbits/mo) so that TFR(I) = DN/25.

The computer equations for TFR were as follows:

```
AN     = RP(I)/TP(I)  (ratio of rabbits/tigers in Ith month)
X      = ( (-1.0)*ALOG(2.0)*DT)*TP(I)/40 (X=exponent)
EXPO   = EXP(X)
DN     = AN*(1.0-EXPO)  (DN = number of rabbits eaten per tiger)
TFR(I) = 2.0*(DN/TFN)  where TFN = 50.0 rabbits/tiger
```

4. Tiger-birth-rate-from-food-multiplier TBFM

The availability of rabbits for tiger food is assumed to have a substantial effect on birth rate, especially in the critical region where the tiger population is undernourished. The graph in Figure 4 shows the relationship between tiger birth rate and the tiger food ratio (TFR) that has been assumed in this model. This is the same relationship used by Forrester for human population in his global model.[3] At TFR = 1 conditions are "normal" and each tiger eats 25 rabbits/mo. A TFR of 2 would mean that a tiger eats twice as many rabbits as normal i.e., 50 rabbits/mo. Notice that the tiger-birth-rate-from-food-multiplier (TBFM) remains constant when TFR is greater than or equal to 4. This reflects the assumption that tigers will not eat more than 100 rabbits/mo. even if they are able to capture more. In other words, they become glutted.

The birth-rate-from-food-multiplier equals the square root of the tiger food ratio for all ratios less than 4. If the food ratio is equal to or greater than 4 the birth rate from food multiplier is 2.

5. Tiger death rate TDR

The tiger death rate is part of the negative-feed-back loop shown in the flow diagram. Under normal conditions the death rate depends on the product of the tiger population (TP) and the normal death rate (TDN). If either the tiger or rabbit population changes the amount of available food per tiger varies and the death rate will change according to the value of the tiger-death-rate-from-food-multiplier (TDFM) which is explained in the next section.

The computer equation is given by

```
TDR(I) = TP(I)*TDN*TDRM

Where:
TDR(I) - number of tigers dying in the Ith month
TP(I)  - tiger population at the beginning of the Ith month
TDN    - normal tiger death rate when there are 2000 rabbits
         and 40 tigers on the island
TDFM   - tiger-death-rate-from-food-multiplier
```

6. Tiger-death-rate-from-food multiplier TDFM

If the food per tiger drops toward zero the death rate must rise steeply. However, after the tigers are glutted with rabbits the tiger death rate will not change as more rabbits are added to the system. The graph in Figure 5 shows the relationship assumed for the tiger death rate (TDFM) and availability of rabbits (TFR). This is the same relationship assumed by Forrester for human population in his global model.[3] Figure 5 shows the tiger death rate decreasing by a factor of 1/2 when the available food for tigers is doubled.

7. Tiger quality-of-life TQL (or Q on graph)

The tiger quality of life is just proportional to the average number of rabbits a tiger is able to capture in a month. (See section 3 on the tiger-food-ratio (TFR) for a discussion of how the number of rabbits/mo. captured by each tiger (DN) is computed.) Like TFR, DN depends on the rabbit and tiger populations during a given month, so that TQL(I) = DN/25.0.

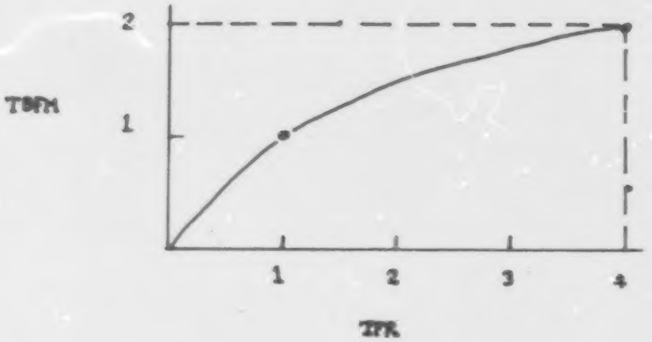


Figure 4: The Tiger-birth-rate-from-food multiplier (TBFM) as a function of the Tiger-food-ratio.

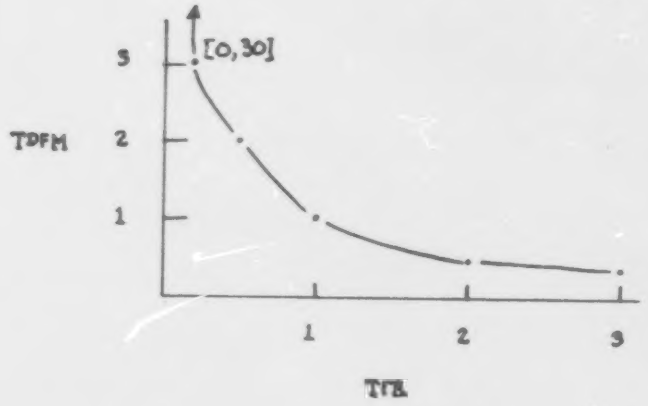


Figure 5: The Tiger-death-rate-from-food multiplier (TDFM) as a function of the Tiger-food-ratio (TFR)

Where:
TQL(I) - tiger quality of life in the I^{th} month
DN - number of rabbits each tiger eats in the I^{th} month
25.0 - number of rabbits each tiger eats in a "normal" month
(Note that TQL = 1 in a "normal" month where tigers are still "hungry")

8. Rabbit Population RP

The rabbit population at the beginning of any month is calculated as the population at the beginning of the previous month, plus the rabbits added by the monthly birth rate, minus the rabbits removed by the monthly death rate. The monthly rabbit death rate describes rabbits that die of "natural" causes such as old age and disease as well as rabbits that are eaten by tigers during the month.

The computer equation for the rabbit population is given by

RP(K) = RP(I) + DT* (RBR(I) - RDR(I))

Where:
RP(K) - rabbit population for month K = I + 1
RP(I) - rabbit population for month I
DT - time interval = one month
RBR(I)- rabbit birth rate during the I^{th} month
RDR(I)- rabbit death rate during the I^{th} month

9. Rabbit Birth Rate RBR

The birth rate is part of the positive feed back loop shown in the flow diagram. Under normal conditions the rabbit birth rate depends on the product of the rabbit population (RP) and the normal birth rate (RBN). If the rabbit population varies and hence the food per rabbit changes, the birth rate changes according to the rabbit-brith-rate from food multiplier (RBFM) which is explained in the next section. As is the case for tigers, the time delay in birth rate due to the rabbit gestation period is neglected.

RBR(I) = RP(I)*RBN*RBFM

10. Rabbit-Birth-rate-from-food multiplier RBFM

This multiplier is computed in exactly the same way as the tiger-birth-rate from food multiplier (TBFM) described in section 4. When there are 2000 rabbits on the island the rabbit food ratio is one and the rabbit birth-rate from food multiplier RBFM is one also.

RBFM(K)=SQRT (RFR(K)) if RFR (K)≤4

The birth-rate-from food multiplier equals the square root of the rabbit food ratio for all ratios less than 4. If the food ratio is equal to or greater than 4 the birth rate from food multiplier remains at 2.

11. Rabbit food ratio RFR

Assume that the island produces a fixed amount of green plant food each month. The rabbit food ratio (RFR) is assumed to be equal to 1 under normal conditions (2000 rabbits). If the population of rabbits doubles, the food per rabbit is cut in half, etc. The equation is then

RFR(I) = 2000/RP(I)

Where:
RFR(I) - rabbit food ratio in the I^{th} month
2000 - rabbits population under normal conditions
RP(I) - rabbit population in the I^{th} month

12. Rabbit Death Rate RDR

The rabbit death rate is due to both natural causes (disease and old age) and consumption by tigers. It is given by the equation

$$RDR(I) = RP(I) * (RDN1*RDFM + RDN2*RDTM)$$

Where:
RDR(I) - no. of rabbits dying in the I^{th} month
RP(I) - rabbit population at the beginning of the I^{th} month
RDN1 - rabbits/mo. dying of natural causes under "normal" conditions
RDFM - rabbit death rate from food-multiplier
RDN2 - rabbits/mo. eaten by tigers under "normal" conditions
RDTM - rabbit death rate-from-tiger multiplier

13. Normal rabbit death rate by "natural" causes RDN1

This death rate describes the rate at which rabbits die from old age and disease under normal circumstances where there are 2000 rabbits on the island. RDN1 gives the normal percentage of rabbits dying each month.

14. Rabbit-death-rate-from food-multiplier RDFM

This death rate multiplier depends on the rabbit food ratio (RFR) in exactly the same way that the tiger death rate from food multiplier (TDRM) depends on the tiger food ratio (TFR). (See section 6 for details.)

15. Normal rabbit death rate (tigers) RDN2

This gives the percentage of rabbits eaten each month under normal conditions (2000 rabbits and 40 tigers).

16. Rabbit-death-rate-from-tiger-multiplier RDTM

This multiplier is given by twice the total number of rabbits eaten/mo. divided by the number of rabbits.

$$RDTM = 2.0 * DN*TP(I)/RP(I)$$

Thus under normal conditions DN = 25 rabbits,
TP(I) = 40 rabbits, RP(I) = 2000 rabbits, so that
RDTM = 1.0 as expected.

V. TESTING MODEL BEHAVIOR

An individual using the model is allowed to change any one of six parameters describing initial tiger or rabbit populations and the birth and death rates for the two populations. These parameters are identified and their values for the standard run are listed in Figure 6 which consists of a listing of the general computer instructions. Using a time sharing system the results of each trial can be typed out on a remote terminal in graphical form. Three quantities are graphed as a function of month for 50 months.

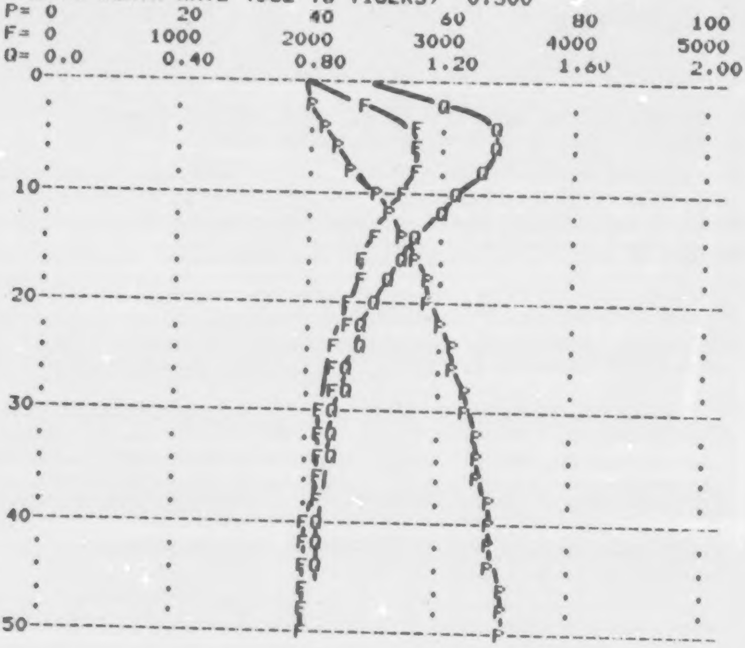
1. P = tiger population (TP)
2. F = tiger food i.e. the rabbit population (RP)
3. Q = tiger quality of life (TQL), i.e., the ratio of the number of rabbits captured each month to the number captured under "normal" conditions.

The Computer output for the standard run and one for a different set of conditions are shown in Figures 7 and 8. Users are usually advised to change only one parameter at a time and are warned that the scale on the graph may change from run to run.

```
RUN TIGERS

   PLEASE ENTER YOUR NAME.

PRISCILLA LAWS

I WILL TYPE THE FOLLOWING INSTRUCTIONS ONCE UNLESS OTHERWISE REQUESTED:


  (A) TYPE 2 TO CHANGE THE INITIAL RABBIT POPULATION
  (B) TYPE 3 TO CHANGE THE NORMAL RABBIT BIRTH RATE
  (C) TYPE 4 TO CHANGE THE NORMAL RABBIT DEATH RATE'(NATURAL)
  (D) TYPE 5 TO CHANGE THE INITIAL TIGER POPULATION
  (E) TYPE 6 TO CHANGE THE NORMAL TIGER BIRTH RATE
  (F) TYPE 7 TO CHANGE THE NORMAL TIGER DEATH RATE
  (G) TYPE 8 TO HAVE THESE INSTRUCTIONS RETYPED
  (H) TYPE 9 TO TERMINATE PLAYING ISLAND DYNAMICS
  (I) TYPE 1 TO RUN THE PROGRAM

HI PRISCILLA LAWS

RIGHT NOW I AM ASSUMING THE FOLLOWING ABOUT TIGERS AND RABBITS ON A
TROPICAL ISLAND

INITIAL RABBIT POPULATION  2000.          INIIAL TIGER POPULATION   40.
NORMAL BIRTH RATE  0.800                  NORMAL BIRTHRATE  0.040
NORMAL DEATH RATE(NATURAL)  0.100        NORMAL DEATH RATE  0.028
NORMAL DEATH RATE (DUE TO TIGERS)  0.500

  TYPE DESIRED INSTRUCTION NUMBER. 9 WILL TERMINATE THIS PROGRAM.
```

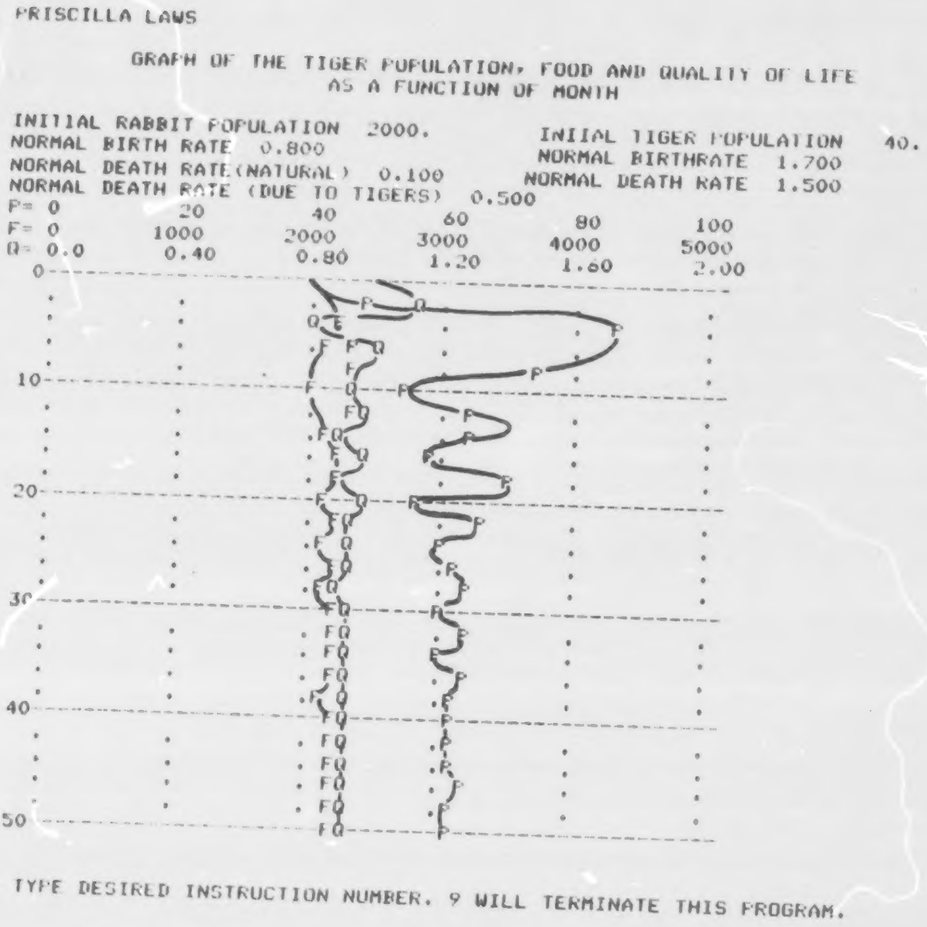
Figure 6: Listing of Computer generated instructions for making variable changes.

```
PRISCILLA LAWS

                GRAPH OF THE TIGER POPULATION, FOOD AND QUALITY OF LIFE
                                AS A FUNCTION OF MONTH

INITIAL RABBIT POPULATION  2000.            INITIAL TIGER POPULATION   40.
NORMAL BIRTH RATE  0.800                     NORMAL BIRTHRATE  0.040
NORMAL DEATH RATE(NATURAL)  0.100          NORMAL DEATH RATE  0.028
NORMAL DEATH RATE (DUE TO TIGERS)  0.500
```



```
TYPE DESIRED INSTRUCTION NUMBER. 9 WILL TERMINATE THIS PROGRAM.
```

Figure 7: Sample Output - The Standard Run.



```
PRISCILLA LAWS

          GRAPH OF THE TIGER POPULATION, FOOD AND QUALITY OF LIFE
                         AS A FUNCTION OF MONTH

INITIAL RABBIT POPULATION  2000.          INIIAL TIGER POPULATION   40.
NORMAL BIRTH RATE  0.800                  NORMAL BIRTHRATE  1.700
NORMAL DEATH RATE(NATURAL)  0.100        NORMAL DEATH RATE  1.500
NORMAL DEATH RATE (DUE TO TIGERS)  0.500
P= 0          20          40          60          80          100
F= 0         1000        2000        3000        4000        5000
Q= 0.0        0.40        0.80        1.20        1.60        2.00
```

```
TYPE DESIRED INSTRUCTION NUMBER. 9 WILL TERMINATE THIS PROGRAM.
```

Figure 8: Sample Output - Oscillatory behavior.

VI. CURRICULAR USES

This model has been used during the past five years in courses in both environmental studies and mathematics.

Since 1972 the Model has been used with four different groups of about 40 students each in an "Environment, Culture, and Values" course[1] that satisfies a humanities distribution requirement at Dickinson. This course was team taught by a Philosopher and a Physicist. Students in this course worked at terminals in groups of two and observed changes in behavior of the tiger/rabbit ecosystem with succesive changes in the input parameters. The student teams were asked to consider the following questions:

1. After 30 or so months the P, F, and Q curves stabilize indicating that the tiger and rabbit populations have reached equilibrium. Which factors influence the final equilibrium? Which do not? Why?
2. What changes in input parameters increase the tiger quality of life the most after the system reaches equilibrium? Why?
3. Assume that we want to optimize the tiger quality of life. What is the equilibrium number of tigers that the island can support under optium conditions? Does this tell you anything about human populations on a finite biosphere?
4. What policies need to be set on the island so that the quality of life can be maximized in the most tigrane (analogous to humane) way? Again can you generalize to human populations on a finite biosphere?

Students reflected on the results of the computer runs in written essays and in class discussion.

The program was used in a similar manner in a year long environmental studies laboratory course for non-science majors entitled "Environmental Science". The Model was used in the fall of 1975, 1976, and 1977 in a unit on Principles of Ecology and Ecological Regulation. The average class size was 45. In this course the instructor invited the author to use one class period to introduce System Dynamics Modeling and the Ecosystem Model to the class. During a laboratory period the students worked out the terminals in groups of 2-3 for about 20 minutes each. A typical group made about 5 runs. The students were asked to maximize the tiger population with a minimum reduction in the tiger quality of life at equilibrium. A class period was then devoted to a discussion of the results obtained by the student groups.

The class discussions tended to be wide ranging. Student groups were asked about their strategies for achieving results. Most groups started by changing the initial tiger or rabbit populations and discovered that these changes had no effect on the equilibrium populations and that only changes in birth and death rates influenced the final populations. Students discussed how this model characteristic might relate to human population problems.

When asked how they measured the "success" on their strategies, students questioned the appropriateness of the definition of the tiger quality of life in the model.

Students also debated how one would measure an optimal equilibrium condition for the ecosystem. Would factors like population x quality-of-life or population + quality of life be appropriate? Would a strategy which maximizes the tiger quality-of-life factor but leaves a small tiger population be preferable to one in which a larger group of tigers experiences a lower quality-of-life?

The use of the model in the mathematics program was more limited. It was demonstrated to several sophomore level classes consisting of about 25 students each in mathematical modeling. The model was also demonstrated to several classes in a senior level games theory course where the optimization of specific parameters were analyzed from the perspective of game theory.

VII. STUDENT RESPONSE

Student reaction to the use of the Model was obtained as part of the course evaluation process in all sections of both of the environmental studies courses. On the basis of these responses and interviews with the course instructors it was felt that the following educational benefits were achieved:

1. A significant number of students in the humanities and social sciences with no previous experience in Mathematical Modeling or Computer use discovered that they "enjoyed the activity and chance to work with the computer system".

2. Students discovered that the model was relatively insensitive to changes in certain parameters. Most students noted that rates rather than initial population levels determined the final system characteristics. They realized an initial tendency to manipulate the system with level changes and wondered whether this "fear of rates" was influencing how policy makers approached real world problems.

3. Students discovered that there are often many different ways to achieve approximately the same equilibrium in the model and that some of these results are counter intuitive.

4. By comparing the behavior of the Model to that of the World 3 model, students were impressed with the similarities between them. They were also impressed with the potential of a greatly simplified model to enhance their understanding of complex real systems and because of the models simplicity, students felt comfortable enough to critize its premises.

5. Students realized that both the development of the model and the uses made of its projections ultimately involve important questions of ethics and fundamental values. For example, in a real ecosystem the tiger-quality-of life (Q) should peak out as the tigers become glutted with rabbits. In this model Q continues to increase indefinately with tiger food. This is more like one conception of Q for human populations in which essentially no limits are set on human desires, i.e., doubling the availability of industrial goods, gourmet foods, etc., doubles the quality of life.

6. Students recognized that the ecosystem represented in the model was greatly simplified and could not adequately describe a real ecosystem. They also realized that the definition of elements such as the tiger-quality-of-life involved value judgements. Thus the students also seemed to have developed an awareness of some of the inherent limitations of Mathematical Modeling.

FOOTNOTES

1 Laws, P.W., "The Physical Scientist in an Interdisciplinary Course". American Journal of Physics, Vol. 41 (June 1973) pp 808 - 812.

2 Forrester, J.W., Principles of Systems. (Wright-Allen Press, Cambridge, Mass., 1968).

3 Forrester, J.W., World Dynamics. (Wright-Allen Press, Cambridge, Mass., 1971) ch 1-3.

4 Meadows, Meadows, Randers, and Behrens, *The Limits to Growth*. (Universe Books, New York, 1972).

5 Meadows, Behrens, Meadows, Naill, Randers and Zahn, *Dynamics of Growth in a Finite World*. (Wright-Allen Press, Cambridge, Mass., 1974).

6 Huntington II Simulation Program - LIMITS (DEC Curriculum Material, Maynard, Mass., 1974).

7 Chester, M., *Graze: Ecology Simulation* (Hewlett-Packard Computer Curriculum Project, 1974).

8 Kerster, H., et al, *Fox and Rabbit Ecology Model*, (Biology Curriculum Group, University of Illinois, 1972).

A LENS AND MIRROR SIMULATION PROGRAM

Robert L. Siddon

United States Naval Academy
Annapolis, Maryland 21402

An interactive computer graphics program has been developed for the TEK4051 that calculates and draws the ray diagram for an arbitrary number of spherical lenses or a single spherical mirror.

The user has the option of specifying:

(1) The type of light source
  (a) Point source at (x,y)
  (b) Parallel rays at angle θ

(2) The location and radius of curvature of each lens surface or the mirror surface

(3) The index of refraction of each lens and the medium to the left and right of each lens

(4) The number of rays plotted in the ray diagram

(5) Whether to plot dashed lines showing the apparent origin of the emerging rays.

The user specifies his problem and plots the ray diagram by a series of commands to the TEK4051. After the plot has been made the user may change any or all of the parameters in the system and plot another ray diagram with the new parameters. A typical plot takes about 1.5 seconds.

The program is used extensively by the Physics Department at the Naval Academy in the second semester of the General Physics with Calculus course (600 students/year). At the present time the Naval Academy has approximately 40 of the TEK4051's for use by the midshipmen.

Examples of use of the program are shown in figures (1)-(4).

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

92

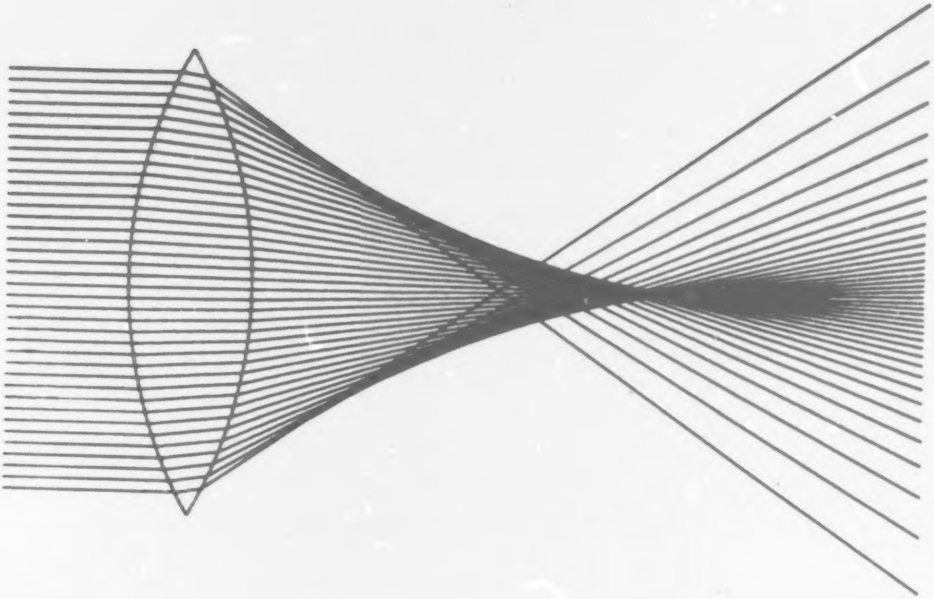


Figure 1. The double convex lens with parallel rays.

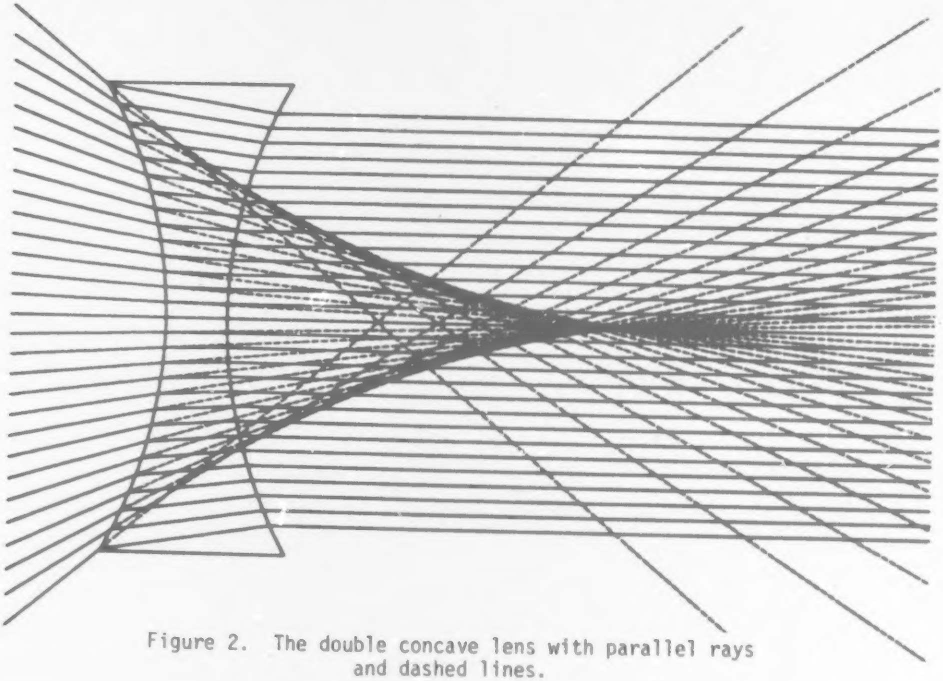


Figure 2. The double concave lens with parallel rays and dashed lines.

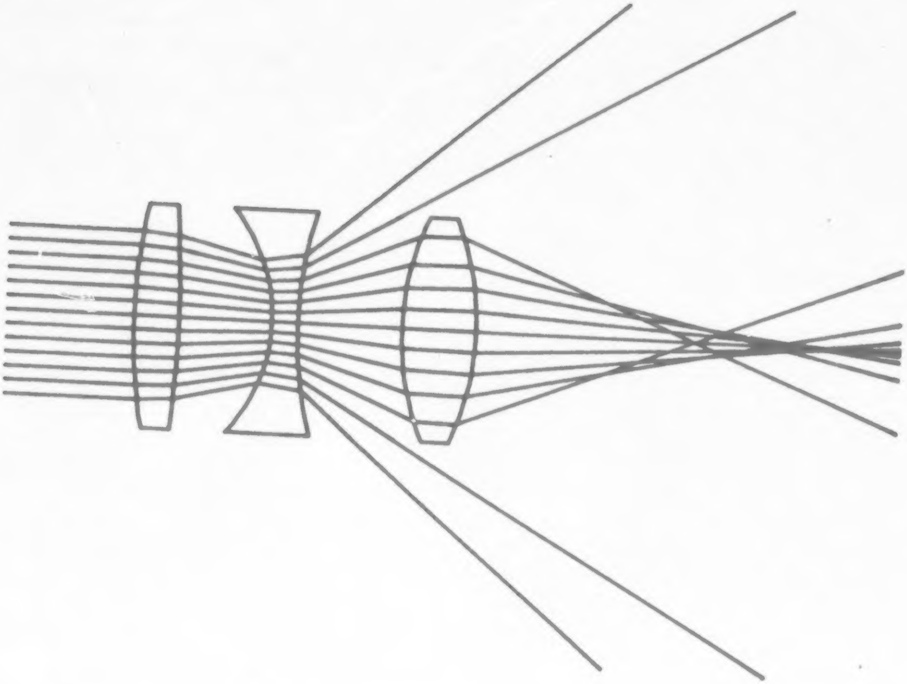


Figure 3. A system of three lenses.

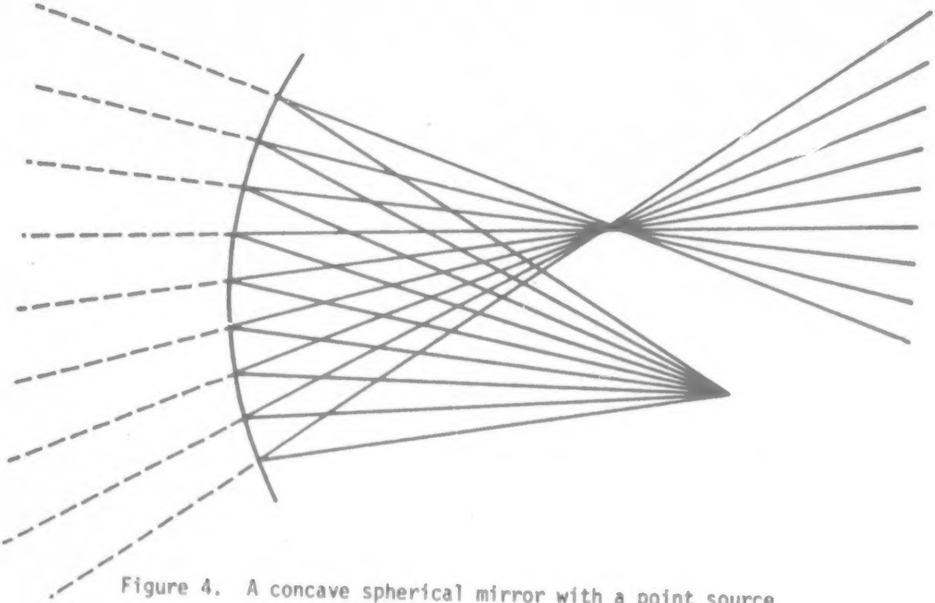


Figure 4. A concave spherical mirror with a point source showing the formation of a real image.

THE USE OF PEERRATE: A COMPUTERIZED SYSTEM OF STUDENT PARTICIPATION IN GRADING

Anthony D. Lutkus, Ph.D.

Rutgers University - Newark

ABSTRACT

The use of a computerized system is described in which students rate each other's term papers. The computer program (PEERRATE) creates a matrix of intervals between ratings received and calculates final ratings incorporating a mean and standard deviation selected by the instructor for the class. The program compensates for the tendency of students to give high or low grades to everyone. Besides computing a norm-adjusted score for performance by each student on each assignment, the PEERRATE program analyzes both the quality of the ratings by raters for that assignment and the consistency of each student as a rater. Follow up survey data indicate that the majority of undergraduates involved endorse the system as used in term paper grading and feel that their learning experience is enhanced.

Can undergraduates with disparate backgrounds meaningfully grade each other's college assignments? This paper will describe a computer system developed at Rutgers University-Newark that facilitates student ratings of each other's work. A basic assumption underlying the development of this system is that students will have an enriched learning experience if they participate in specifying grading criteria and in evaluating student assignments according to those criteria.

This report is in three sections: First, a brief background of the PEERRATE computer program; second, a consideration of how the system has been used in undergraduate courses; and third, preliminary results from PEERRATE and a

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

summary of student reaction gleaned from surveys taken at the end of courses which employed the system.

BACKGROUND OF THE SYSTEM

Over the past few years, MBA students at Rutgers-Newark Graduate School of Business have been rating each other's performance on team projects for business corporations. Inherent in any rating task where the participants rate only a subset of the total group is the possibility of individual rater bias (either too lenient or too stiff) distorting the ratings. In addition, there is the human problem of students tending to want to give each other an "A" and, consequently, using only the high end of any rating scale. To solve these procedural difficulties, Dr. Purnell Benson developed (1976, 1977) the PEERRATE program which takes student-provided ratings as input, reduces rater bias, adjusts the rating distribution to a pre-set mean and deviation, and then prints a report for each participant including his calculated grade and his efficiency as a grader.

The most obvious solution to the rater bias problem would be to have raters rank all performers from first to last, thus avoiding the "all-A" problem. This means, however, that every student rater has to rate every performer -- an impractical situation in a large class. One can, however, ask raters to rate a subset of the class on a scale with behaviorally defined categories like "excellent", "very good", "fair", or "outrageous." The point intervals or steps between different performers can then be established. Now everyone need not rate everyone else. The size of the intervals between rated performers establish the differences in their achievement.

Even if a class of raters tends to use only the upper end of a rating scale, the intervals between performers remain. Thus, it is a simple algebraic maneuver to spread out the part of the scale that has been used, maintaining the rank orders between performers but setting a new mean and deviation. The average score given by the raters sets the achievement norm for the entire group. An absolute measure of achievement can be designated by an outside authority (usually the teacher!) who knows what the group as a whole has produced.

The PEERRATE program solves the problem of incomplete sets of rating intervals by using Mosteller's technique (1951) of finding the best set of final ratings which agree most closely with the inputed rating data. It is a least squares procedure in which the least errors of inconsistency remain between the calculated difference separating performers and the original differences reported by the raters.

Other procedural problems to be solved before peer rating could be made practical included: providing an evaluation of rating skill so students will take the rating task seriously, finding a way for the teacher to detect faulty or discrepant ratings so they can be adjusted, and, finally, providing anonymity so that raters cannot favor their friends.

Dr. Benson's PEERRATE program automatically outputs a measure of the rater's skill in rating. It is possible, then, for the instructor to assign a "grade for grading." Four separate but related measures of rating consistency are used. First, the individual grades given by each rater are compared with the grades given to the same performers by the class as a whole. The latter, general grades are given by computer calculation and are called "adjusted group ratings," or AGR's. Thus, the AGR's provide one standard for evaluating raters. The instructor can substitute his own grading on the assignment for the AGR's if he wishes. In fact, the program can include the instructor's grades along with student'generated grades and weight each according to whatever proportion the instructor deems appropriate.

The second evaluation of rater skill addresses the average bias of each rater. Does he or she rate too high or too low in general? The computer reports a "bias" score for each rater. Thirdly, the range of the rating scale that the rater uses can provide an indicator of his rating skill. Does he use more or less of the scale than the entire class? Again, the computer produces a difference score between the individual rater's range and that of other raters.

Fourth, after adjusting the grades for systematic bias and for the range used, the computer prints the standard deviation of the residual error that remains between the rater's ratings and the AGR's. If students also serve as raters for their own work, the discrepancy between self-rating and class rating can also be computed. These measures of rating skill are combined by the computer into a single score for rating ability. This composite rating ability score could conceivably serve as one component in the student's course grade.

Aside from producing a measure of each rater's skill at rating, Benson's program makes it possible for the instructor to assess the quality of all the ratings that went into a single performer's Adjusted Group Rating. AGR's with low agreement can then be reviewed by the instructor and possibly changed. PEERRATE prints a table of rank ordered scores for ratee performance, quality of ratee scores, and rater ability. The instructor can review the individual work involved to see whether, in particular, low grades with low agreement are justified. If the grades with low agreement are reasonable, then the instructor can assume that the AGR's for the rest of the class are valid. The instructor can review, for instance, a sample of term paper assignments beginning with the low AGR's, some middle level scores, and a few at the top of the distribution. If the papers seem out of line with the absolute score value that has been set for the class norm, the batch can be run again with a different norm. Alternatively, it may be that the class was either not properly prepared for the grading task or did not absorb sufficient course content.

USE OF PEERRATE AT RUTGERS-NEWARK

The PEERRATE program is adaptable to several types of class assignments. Written essays, oral reports, term reports, team reviews, or co-written tests can each be handled by the program. My own use of the system has focused on grading term papers in large classes of introductory and child psychology. It has been my experience that one can assign long term papers in large classes (normally taboo to an over-worked instructor) by using PEERRATE to assist in the grading process.

Previously mentioned problems of anonymity and avoidance of peer favoritism can be handled by classroom procedures that are almost as important as the computer program itself. Students hand in multiple copies of their assignment, one for each rater. Computer identification numbers are assigned to each student and all identifying information is then removed from the papers. In a subsequent class, each student receives a package of papers to grade (usually between 3 and 6), an answer sheet for the grades, and, in our instance, a questionnaire to complete regarding the system. Student numbers are assigned from 2 (number 1 being reserved for the instructor's grades, if used) to N+1, with N being the class size. The program is not affected by extra numbers or missing numbers because it automatically contracts the matrix size during the calculations. At present, 35 is the maximum number of raters that can be inputted. Larger classes can be served by splitting them and treating the subsections independently.

The answer or rating form is prepared by the instructor. Appendix A displays a sample of one used in my courses. The student rater's name is included for ease of distribution but the ratee's names remain anonymous.

A feature that makes PEERRATE a useful pedagogical tool is the provision for multiple ratings, i.e., one for each of several criteria. I have found that it makes for a stimulating class exercise to have students develop the criteria themselves. When the assignments are term papers, the criteria selected have followed nearly the same pattern over four semesters: content, organization, grammar or style, originality, and overall worth. In addition, the relative weights of each of the criteria can be set for each run of the program. Students enjoy debating -- at length -- whether, for example, content and organization should be of equal weight with grammar, style, etc. From the instructor's point of view, such thought about the prerequisites of a good paper can only be beneficial. The instructor then explains his interpretation of the criteria, possibly gives examples and then includes brief written descriptions (see Appendix B) of the criteria at the time the papers are distributed for rating. The program can accept any range of scales so it is possible to use, say, a 0 to 20 scale instead of the standard 1 to 100. The purpose of this is to expand the range of the scale students tend to use. If 1 to 100 is used, students tend to think of 70 as "passing." The wider the spread of scale intervals employed by raters, the better will be the discrimination among ratees. On a scale of 1 to 20 or 1 to 35, the preconceived "anchor" of "70" is harder to define and the rater tends to rely on the category labels attached to the scale by the instructor.

When a given term paper is rated by less than the entire class, the numerical assignment of ratees to each rater becomes extremely important. The subsequent paired-comparison calculation by PEERRATE assumes that there is, in Benson's words, "a distributed, balanced design." "Distributed" refers to the requirement that each student be rated by a different subgroup of peers, i.e., the combination of rater identification numbers is unique for each rater. "Balanced" means that each subgroup of raters is scattered through the class distribution, not bunched into clusters. The assignment of the subgroup of ratees for the first student rater (#2) would proceed as follows (Benson, 1976):

> "...Then students to be rated by the lowest rater, 2, are selected by taking subscript numbers which fall proportionately from the largest ID number, say 30. Using 2/3 as the proportion, the ID numbers selected for six ratees, rounding to the nearest integer would be 20, 14, 10, 7, 5, 3. This same configuration is extended to other raters by increments. If the rater's ID number is, say, 20, then add 18 to each in this configuration, yielding 9, 3, 28, 25, 23, and 21."

More detailed examples of ID number assignment procedures will be available in a PEERRATE handbook that is in preparation (Benson, 1978).

Once the multiple copies of student papers are received, the instructor or an assistant assigns the ID numbers to the papers and collates the package of papers to be returned to each rater at the next class meeting. Students can either do the rating in class or take the papers home for a more leisurely reading. The rating sheets are returned, along with the term papers with the comments of the rater. If PEERRATE is used near the end of the semester, sufficient time must be allowed for the students to rate the papers, return them, and then for the instructor to run the program and return the printouts. A week is usually sufficient time for rating, computer run, and the return of the papers and printouts.

Aside from keypunching the ratings, a PEERRATE user must prepare a problem card which controls program parameters and options, plus a list of student names that will appear on the printout. Table 1 presents Benson's outline of the problem card. Notice that the problem card would contain the mean and standard deviation that the instructor selects for his class.

RESULTS FROM PEERRATE AND STUDENT REACTIONS

If students are properly briefed on the rating task and if the class as a whole has gained some knowledge of the subject matter, I have found the adjusted class ratings to be a good estimate of the rank orders I placed on the papers when reading them myself. PEERRATE routinely prints a matrix of inter-correlations of all the variables in the system. The correlations obtained between paper grades and the writer's ability as a rater ranged between $r = .632$ and $r = .703$ ($P < .01$). This is suggestive of internal validity of the technique since students who write good papers show up as better paper raters as well.

Table 2 presents a sample of the printout the instructor receives reflecting the rank orders of the entire class by paper grade (performance score), quality of rating for each paper, and, finally, the rater's ability as rater. This table serves as an important diagnostic for the teacher before the return of the student printouts. If the instructor chooses not to read all the papers, he can still supervise the accuracy of the performance scores (paper grades) by reading a subset of papers made up of: (a) papers with low grades; (b) papers with high grades; and (c) papers with low rater quality scores, especially if they coincide with low performance grades. Low rating quality scores (the middle set of columns in Table 2) indicate rater disagreement about the paper after the calculation of adjusted class ratings. Examples of papers that would be checked in Table 2 would be ID numbers 9, 28, 20, 36, 19 (papers with low scores and low rater agreement); number 22 (high score, low agreement); and 39, 15, and 16 (high scores). Should the instructor disagree with the paper's grade, he simply overrides it -- either on the student's printout or by indicating the change on a subsequent run of the program.

A second level of pedagogical diagnosis occurs in the student's printed report. A sample is included in Table 3. For this student, the overall paper grade is 87 (upper right hand corner). The five criteria used in grading are listed next, with their respective norm adjusted scores: content, organization, writing style, originality, and overall. Notice that the pattern of these scores can indicate to a student where the relative strengths and weaknesses of his paper lie. The student in Table 3 has a good score for content and originality but his organization and writing style (grammar) suffered in comparison. The end column in this section of Table 3 lists the weight given each of the criteria in the computation of the adjusted class rating (the 87 in this case).

The student can also see the adjusted ratings given his paper by each rater on each of the criteria. These are displayed in the middle of Table 3 "Report of Ratings Received from Individual Raters." The first row of numbers from each rater (item ratings 1 through 5) contains the ratings on content, organization, etc. The next three rows contain measures of that rater's bias. High negative numbers indicate a stiff grader; high positives a lenient one. The closer to zero the numbers appear, the closer the rater came to the class rating characteristics. "Rater residual" refers to the standard deviation of the residual error remaining in the rater's ratings after his bias was adjusted by the program.

The final section of Table 3 displays measures of this student's rating ability. The top score "Overall Score for Rating Performance" is a composite measure made up from the next four measures of rating agreement (defined earlier). The instructor sets not only the mean and deviation of the student's paper grades, but also the mean and deviation of the distribution of student scores for rater performance. As mentioned earlier, the score for rating can then also become a part of the student's grade on the assignment as a whole. In the class example used here, the mean was set at 85 and the deviation at 7, thus 2/3 of the class scores fell between 78 and 92. The paper assignment was worth 50 points (not 50%) in the grading scheme for the course, so the students could calculate their point score on the paper by dividing their overall performance score in half. PEERRATE can print results on any point scale and transform one scale into another. The overall score could easily have been printed on a 0 to 50 scale.

For the past four semesters, an anonymous questionnaire has been distributed following the peer rating exercise. The question form solicited comments from student users and also asked specific questions about degree of interest and the amount of learning the students felt PEERRATE added to the course. Table 4 presents the results of the survey. The responses to the last two questions in Table 4 suggest that the majority of students feel they learn more by reading each other's papers and an overwhelming majority (79%) feel that PEERRATE is at least worth further development in conjunction with the course taken.

Was the questionnaire a sensitive instrument or did it suffer from student social compliance with a project initiated by the instructor? Further analysis of the classes making up the survey data showed that one could divide them into groups that had filled out the questionnaire after rating but before grade feed-back and those who responded after rating and receiving printed results. The percentage of responses "yes" to the "continue" question was unchanged. However, the percentage in the other two categories shifted significantly ($\chi^2 = 18.015$, df = 2, $p < .01$), with more "no's" in one post-feedback group. It is to be expected that some students will always be dissatisfied with any grading system and certainly PEERRATE is not a grading panacea. The data indicate that the students did react honestly. The majority response to the "learn more" question alone suggests that the system has didactic as well as pragmatic value.

Students tend to react more positively to the system if they feel the instructor is doing his "share" of the work and if they can have input in the form of setting initial criteria and including a "self" rating along with other raters. The difference between self ratings and the group adjusted rating becomes another measure of rating skill. In one class of 77 students, the raw score mean rating for the class and the raw score mean self rating differed by about two points, a difference that was non-significant by t-test.

When the PEERRATE system is first announced in a class, it sometimes raises questions. Some students initially feel they are not competent or well enough informed to rate each other. This is especially true in classes like child psychology which attract students from a variety of majors and class years. While PEERRATE might be most suited to homogenious upper class courses in a given major field, the instructor can structure class assignments so that it is amenable to more heterogenious classes. One strategy that has worked at Rutgers is to assign a term paper that is based on a set of readings that is common to the class. For example, I have assigned a book of secondary source readings and required term papers that traced the primary source materials underlying a chosen subset of the readings.

Experiments assessing the objective amount of learning increase attributable to PEERRATE (rather than student opinion) are in the planning stage. The peer rating system described here, however, would seem to have benefits beyond learning the content in a given course. The experiences of "sitting on the other side of the professor's desk," of participating in the definition and analysis of rating criteria, and of constructing a paper with multiple readers in mind assist in developing the kind of independent, critical thought processes that are the goals of a liberal education.

PEERRATE AVAILABILITY

PEERRATE is written in FORTRAN. It consists of over 4,000 input cards, many of which are comments and diagnostics to aid the user. The program is currently mounted on the New Jersey State Educational Computer Network (IBM 370) and is available to the public. It can be run on any medium-sized computer. Storage requirement is about 240K. The typical user in higher education need only keypunch his system's JCL cards, the PEERRATE problem cards, and his student's input ratings.

REFERENCES

Benson, Purnell H. "A Computerized System of Student Grading of Student Assignments." *Third Annual New Jersey Conference on the Use of Computers in Higher Education*. New Brunswick, New Jersey: New Jersey Educational Computer Network, Inc., 1976, 21-30.

Benson, Purnell H. A Supervised Student Grading System. *American Statistical Association, Social Statistics Section*, 1977 (in press).

Benson, Purnell H. *Documentation for the Use of PEERRATE*. Rutgers University Center for Computing and Information Services, Newark, New Jersey, 07102.

Mosteller, Frederick. Remarks on the Method of Paired Comparisons: The Least Squares Solution Assuming Equal Standard Deviations and Equal Correlations. *Psychometrika*, 16, 1951, 3-9.

TABLE 1
INPUT PARAMETERS FOR PROBLEM CARD AND OTHER CONTROL CARDS

1. Problem card.

Parameter

Highest ID no. used for student in class
Number of tasks rated
Number of sets of records
Exponent for skewing rater scores
Grade norm imposed by instructor
Grade stretch imposed by instructor
Section number of class
Multiplier for deviation in rater score
Multiplier for variability in rater score
Origin for deviation in rater score
Origin for variability in rater score
Minimum number of ratings for inclusion
Minimum permissible grade
Maximum permissible grade
Enter 1 if instructor grade imposed to override class rating
Enter 1 if grade is average of tasks, not sum
Enter 1 to print matrixes, 2 for solution, 3 for both
Enter 1 for printout of individual grade sheets
Number of iterations if rater weights calculated

2. Point spread for tasks (exercises or assignment parts).

3, Weights for tasks.

4. Rating norms and rating spreads for tasks.

5. Weights for raters, or parameters for iterating weights.

6. Number of student ratings inputted by rater in each record.

7. Names of tasks.

8. List of names in class, ID Nos., and 1 punch if instructor grade imposed.

9. Rating records supplied by students, and by instructor if acting as a student.

10. Instructor's imposed grade and grade weight. (optional)

For column assignments for input 2 through 10,
consult the particular version of program used.

TABLE 2

PERF SCORES, RATING QUALITIES, AND RATER ABILITIES IN ASCENDING ORDER

PERF SCORE			RATING QUALITY			RATER ABILITY		
ID NO.	SUB NO.	SCORE	ID NO.	SUB NO.	SCORE	ID NO.	SUB NO.	SCORE
9	9	63	9	9	67	19	19	****
28	28	64	36	36	68	21	21	****
20	20	70	12	12	76	36	36	68
36	36	71	28	28	78	20	20	72
19	19	74	20	20	79	33	33	73
27	27	74	7	7	79	9	9	74
41	41	74	19	19	79	28	28	76
3	3	76	26	26	81	38	38	76
12	12	77	3	3	81	3	3	78
24	24	77	21	21	81	12	12	79
13	13	78	24	24	82	7	7	82
23	23	80	22	22	82	24	24	82
37	37	81	10	10	83	41	41	82
40	40	81	37	37	83	16	16	85
33	33	81	41	41	83	6	6	85
31	31	82	33	33	84	13	13	86
7	7	82	31	31	84	37	37	86
4	4	83	39	39	85	26	26	86
32	32	83	38	38	86	34	34	86
14	14	83	40	40	87	22	22	87
34	34	83	23	23	87	32	32	87
26	26	84	11	11	89	17	17	87
11	11	85	6	6	89	11	11	87
38	38	85	13	13	89	4	4	88
17	17	86	16	16	89	29	29	88
5	5	87	17	17	89	23	23	88
21	21	87	34	34	89	10	10	89
8	8	87	15	15	90	2	2	89
6	6	87	8	8	90	39	39	89
30	30	88	14	14	90	14	14	89
18	18	88	32	32	90	5	5	90
2	2	90	2	2	91	27	27	90
1	1	90	5	5	91	15	15	90
35	35	90	35	35	91	31	31	91
29	29	90	29	29	92	30	30	92
10	10	91	27	27	93	25	25	92
25	25	92	4	4	94	8	8	93
16	16	93	30	30	95	40	40	94
15	15	94	18	18	95	35	35	94
22	22	95	25	25	97	18	18	95
39	39	97	1	1	99	1	1	95

TABLE 3
A SAMPLE STUDENT PRINTOUT

```
CALLARI,L.L.   ID NO. 6   PROJECT 1277   NUMBER 1   *PEER-RATING*
PROF. ANTHONY LUTKUS           DECEMBER 1977       #B

   **** OVERALL SCORE FOR PERFORMANCE OF ASSIGNMENT(S)---------87
        THIS IS SCORE RECEIVED ON ASSIGNMENT WITH 3 RATERS RECORDS

        POINTS FOR ITEMS ADJUSTED TO NORMS FOR GROUP:
                             POINT     ITEM RATING      WEIGHT
          ITEM NAME          BASIS     ADJ TO NORM     FOR ITEM
        CONTENT               100          91            0.210
        ORGANIZATION          100          80            0.180
        WRITING STYLE         100          84            0.150
        ORIGINALITY           100          90            0.060
        OVER-ALL              100          89            0.400
       SUCCESSIVE ITEM SELF-RATINGS BY PEER ON POINT BASIS ABOVE:
                         90    85    95    95    95
   ****REPORT OF RATINGS RECEIVED FROM INDIVIDUAL RATERS ON POINT
       BASIS ABOVE

  RATER       COMPARED WITH    SCORE        ITEM RATING GIVEN BY RATER
ID NO.  WT.   ADJ. GROUP RTG   GIVEN        1      2      3      4      5
         1    RTGS FROM RTR:    80          85     80     70     90     80
                   RATER BIAS     2.9      -4.6    0.8   -8.5    0.3   -6.9
              RATER RANGE DIF     2.1      -0.7    3.7    5.5   -0.8   -0.5
               RATER RESIDUAL     2.6       2.5    8.9   14.7    3.4    3.2

         1    RTGS FROM RTR:    92          90     85     95     95     95
                   RATER BIAS     3.1       3.1    7.3    7.0    5.2    6.6
              RATER RANGE DIF     2.2      -2.9   -0.8   -2.9   -1.2   -2.6
               RATER RESIDUAL     2.1       5.9    7.2    6.9    4.2    5.1

         1    RTGS FROM RTR:    96         100     95     95     95     95
                   RATER BIAS     3.1       6.3    7.6    7.8    6.2    4.4
              RATER RANGE DIF     2.2       0.7   -0.9    5.4    0.1    1.3
               RATER RESIDUAL     2.1       4.7    7.7   10.7    2.0    3.0
   ****OVERALL SCORE FOR RATING PERFORMANCE----------------------- 92
       THIS IS SCORE RECEIVED FOR RATING PERFORMANCE WITH 4 RATEES

     (CALCULATIONS WHICH FOLLOW ARE DONE ON BASIS OF 100 POINTS)
       SCORE FOR DEVIATION FROM ADJ GROUP RATINGS ----------------- 87
       WEIGHT FOR DEV FROM ADJ GROUP RATINGS ------------ 1.0

       SCORE FOR DIFFERENCE IN RANGE FROM ADJ GROUP RATINGS------ 93
       WEIGHT, DIF IN RANGE FROM ADJ GROUP RATINGS ------ 1.0

       SCORE FOR RESIDUAL ERROR FROM ADJ GROUP RATINGS ---------- 100
       WEIGHT, RESIDUAL ERROR FROM ADJ GROUP RATINGS ---- 1.0

       SCORE FOR ERROR IN SELF-RATINGS FROM ADJ GROUP RATINGS ---- 98
       WEIGHT, ERROR IN SELF-RTGS FROM ADJ GROUP RTGS --- 1.0
          (IF BLANK, NO EVALUATION OF SELF-RATINGS IS MADE)
```

TABLE 4

RESULTS OF PEERRATE FOLLOW-UP SURVEY (N=186)

Question		Percent of Respondents
Clarity of instructions?	Extremely clear	30
	Quite clear	59
	Somewhat clear	10
	Not at all clear	1
PEERRATE make the course interesting?	More	55
	Less	8
	Little difference	37
Amount of learning?	More	64
	Less	4
	Same	32
Continue PEERRATE in this course?	Yes	41
	No	21
	Worth further trial	38

APPENDIX A

NAME Perez, Nereida YOUR PAPER # 8A

PRINCIPLES OF PSYCHOLOGY
Dr. Anthony D. Lutkus

PEER-RATING SHEET

****

SUGGESTED RATING SCALE

Outrageous	Poor	Adequate	Good	Excellent
1-2	3-4	5-6	7-8	9-10

****

1st Paper # 11A

Content 8 Organization 6 Writing Style 8 Originality 7
Over-All 8

2nd Paper # 25A

Content 6 Organization 9 Writing Style 8 Originality 7
Over-All 7

3rd Paper # 39A

Content 10 Organization 10 Writing Style 9 Originality 7
Over-All 10

Rate Your Own Paper # 8A

Content 8 Organization 9 Writing Style 7 Originality 8
Over-All 8

11/77

APPENDIX B

PEER RATING DATA SHEET

***** RETURN RATING SHEET TO CLASS OR TO DEAN'S OFFICE, 325 HILL HALL *****
BY THURSDAY, DECEMBER 8

You will be rating your fellow students' papers on five categories that were discussed in class. You should give your rating on each dimension as independently as possible. The rating score can range from 1 to 10 points according to the guideline below. The ratings will remain anonymous. Remember that the computer program is concerned only with the intervals between the ratings. There is also a provision to adjust the mean of the distribution and the relative dispersion around that mean. Therefore you will not benefit yourself or others by giving falsely high ratings to everyone. Try to use the whole scale and to assess the papers as fairly as possible. Record your ratings in the spaces on the rating sheet.

SUGGESTED RATING SCALE

Outrageous	Poor	Adequate	Good	Excellent
1-2	3-4	5-6	7-8	9-10

THE RATING CATEGORIES

CONTENT: Includes adequacy of the summary of the articles within the page limitations. Are outside sources brought in? That is, the text, lectures or other books, articles from journals. Are several points of view given if the topic calls for them?

ORGANIZATION: Is there some logical ordering to the presentation of the material? If so, can the reader easily uncover the structure? That is, are there outlines, headings, introductory and/or summary statements? Do paragraphs join together coherently? Are different issues, arguments and the data to support them clearly delineated? Does the author build a case using the material he has presented or does he/she merely summarize another author?

WRITING STYLE: Is the paper in literate English? Do subject and verbs agree in number and tense? Are the sentences complete? Is the writing style clear, i.e., not rambling and/or awkardly constructed? Is it a good "composition?"

ORIGINALITY: Does the paper sound suspiciously like a text book? Or does it sound like a student writer? Honest student attempts (with their occasional errors and awkardness) are to be preferred to prose that is too closely paraphrased from published material. Does the writer himself have an opinion? Is it based on the material he has presented?

OVER-ALL: Give your best estimate of the paper's over-all quality. Do not, however, simply take an average of the four categories above.

PROBGEN: AN INTERACTIVE COMPUTER PROGRAM FOR FACULTY USE IN CREATING AND GENERATING INDIVIDUALIZED, NON-IDENTICAL PROBLEM SETS AND TESTS

Ronald W. Collins and Steven J. Duff
Department of Chemistry
Eastern Michigan University
Ypsilanti, Michigan 48197
(313)-487-0106

The computer has been used for a variety of instructional purposes in chemistry education. As shown in Figure 1, these uses range from classical data reduction to sophisticated graphics and tutorial computer-assisted instruction (CAI). Among the more widely-used and more successful techniques is computer-assisted test construction (CATC). According to Lippey[1], the computer can support test preparation in five ways: (1) item banking, (2) item generation, (3) item attribute banking, (4) item selection, and (5) test printing. In addition, the computer can facilitate grading by generating answer keys and/or by actually scoring the exams if the students record their answers on answer sheets which can be processed via an optical scanner. This scoring procedure is, however, basically an administrative function of the computer and as such is not the principal focus of CATC.

In concept, there are fundamentally two different approaches to CATC which have been used in chemistry education. The first is based on the ability of the computer to facilitate information retrieval (hereafter referred to as CATC-R), and the second method involves the computer in a true production or generation role (called CATC-G). In other words, in CATC-R the computer merely processes or manipulates a categorized database of complete questions, with or without answers, but does not actively create or add to the content of the individual questions themselves. This technique is dependent upon the availability of an extensive, high-quality inventory of appropriate questions in machine-readable form ("item banking").

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© 1978 CCUC

In chemistry, the two best known CATC-R systems are those of Moore[2] (based on an earlier system developed by Prosser[3]) and Willis[4]. In each instance there is a main program which controls the computer-based processing of the database of available questions. The questions are classified according to topic, degree of difficulty, etc. It is important to stress once again, however, that only "what goes in, comes out"; i.e., the computer does not solve the problems as it sorts through and randomly selects from the problems as it sorts through and randomly selects from the available categories. Answers will be printed for the tests generated only if the problems were solved manually and the answers had been included in the database. Clearly the major problem in CATC-R is the acquisition of questions, and in response to this need, Willis[5] is currently coordinating a national effort to create a clearinghouse of chemistry questions.

In contrast to this approach, in CATC-G the computational power of the computer is employed to "fill in" values within problems, and to solve the resultant, unique problems ("item generation"). In this instance there is no need for an inventory of questions which will be fed into the machine. Instead, a more sophisticated need now arises, since a unique program or subroutine must be written for each type of problem. Essentially, the author determines the format or "skeleton" of the questions desired, and the proper algorithm is constructed for a program which will then generate values randomly within appropriate limits and "fill in the blanks" in the question. Concurrently, the computer solves the problem and provides the correct answer on the key which accompanies the tests generated. In chemistry, the most successful CATC-G system is that developed by Johnson[6] at the University of Pittsburgh, which currently consists of approximately 400 subroutines (each being an item or question prototype) which are in turn executed and controlled by a main program.

Careful analysis reveals that the implementation and expanding use of either type of CATC system can be extremely time consuming. In the case of CATC-R, considerable time will be spent either in personally writing new questions or in reviewing and/or classifying questions obtained elsewhere. On the other hand, in CATC-G considerable time will be spent in the actual writing of appropriate programs, one per question type. This paper describes a new FORTRAN computer program called PROBGEN which greatly reduces the time associated with using either type of CATC, or which can be used as a stand-alone "pseudo CATC" system.

In essence, PROBGEN is an interactive computer program designed for faculty members without any knowledge of computer programming. Via PROBGEN-directed dialog the user creates, or indirectly "writes" on-line, a program which will in turn produce (essentially CATC-G) in batch mode an endless supply of non-identical problems of the specified type. The efficient use of PROBGEN demands that the faculty member conceptualize the desired problem into a generalized format, and then reduce the mathematical processing involved to a single "answer equation". This information is then transmitted on-line to the computer via a sequential question-and-answer interaction, after which PROBGEN "writes" or assembles the appropriate program. Next, the desired number of problems of the specified type are generated by this second program, and printed in batch mode. Each test or assignment can consist of one or more problems of as many types as are defined or input to PROBGEN.

A diagram depicting the use and operation of PROBGEN is shown in Figure 2. In addition, as shown in Figure 3, PROBGEN could be used in conjunction with CATC-R, as a means of creating sets of analogous, non-identical problems to comprise the database from which the CATC-R program will create the tests. Also, PROBGEN could be used to write the subroutines or programs for use within a CATC-G system.

In outline form, the steps involved in using PROBGEN are as follows: (steps 1-9 off-line)

1. Define the type of problem, including the quantities(A, B, C,...) which will be given, and the quantity (X) whose value will be calculated as the goal of the problem.
2. Using the fundamental equation(s) as defined by the theory underlying the problem, combine and/or arrange as needed so as to isolate an "answer equation(s)" of the form X=f (A, B, C...).
3. Determine whether each quantity (A, B, C,...) will have a fixed, constant value or a randomly-assigned value within the desired problems which are to be generated.
4. For those quantities among A, B, C, etc. which are to have randomly-assigned values, decide, by specifying upper and lower limits, upon a range of values which is consistent with the theories involved, and the objectives of the particular problem.
5. For each quantity, A, B. C, etc., select a FORTRAN code (variable) name which is consistent with its integer or floating point character.
6. Determine any limitations, restrictions, or forbidden relationships among, and/or arising from computations based upon, the various quantities A, B, C, etc.
7. Compose the exact wording of the problem as you wish it to appear on the test or problem set.
8. Repeat steps (1) through (7) for each additional problem type which is to be included on the desired test or assignment.
9. Compose the heading which you wish to have printed on each individual test and also determine how many problems of each type are to be included per test, and how many tests are desired.
10. (On-line) Access PROBGEN via an available terminal and input the appropriate equation, values, and text for each problem type as requested; i.e., respond with the proper input to each question posed by PROBGEN. Check all input when it is printed back, in order to catch typographical errors before they possibly suppress the execution of the program.
11. Go to the computer center to pick up the tests which are output via the high-speed line printer.

Turning to several specific examples in the field of chemistry, Table 1 shows the off-line conceptualization and organization of (1) a limiting reagent stoichiometry problem, and (2) a gaseous equilibrium calculation. The complete dialog with PROBGEN for these two problem types is shown in Figures 4(a) - 4(d) with the user's responses being underlined in each case. A sample problem set or test from the output is shown in Figures 5(a) - 5(b). It is interesting to compare the imput answer equations as shown in Figure 4(a) (marked with an asterisk, *) to those given in Table 1 for the limiting reagent problem. First, for the input to PROBGEN the symbols are coded into appropriate FORTRAN integer or floating point variable names; however, this is relatively easy to follow. Second, and more important, the input equations shown in Figure 4(a) contain far more terms than those given in Table 1. The reason for this is that a variable name must appear in the input answer equation in order for PROBGEN to call for the value, or range, which is to be assigned to that variable. Therefore, whether or not a particular quantity is an integral part of the solution equation yielding the numerical answer, it must be entered as a "dummy variable" within the equation. Normally this is done by including it both in the numerator and denominator, thereby effecting its cancellation. For example, in conventional symbolism, input equation 1 is as follows:

$$\text{mass}_D = \frac{(\text{mass}_A)\ (d)\ (MW_D)\ (\cancel{b})\ (\cancel{c})\ (\cancel{MW_B})\ (\cancel{\text{mass}_B})}{(a)\ (MW_A)\ (\cancel{b})\ (\cancel{c})\ (\cancel{MW_B})\ (\cancel{\text{mass}_B})}$$

Invoking the cancellations as shown, the equation obviously reduces to exactly the same form as given in Table 1. Efforts are currently underway to treat "nonessential variables" in PROBGEN without the necessity of proceeding via this "dummy variable" route.

In this particular example, two problems of each type (stoichiometry and equilibrium) are included on every test or assignment, making a total of four problems as shown in Figures 5(a) and 5(b). Notice, however, that it is not necessary to repeat the lengthy question-and-answer cycle for the second occurrence of a given problem type. For example, as shown in Figure 4(c) the input for the second problem is merely "SAME". Also noteworthy in Figures 4(c) and 4(d) is the procedure for defining the numerical restrictions (i.e., forbidden values) on several quantities (see Table 1).

For the sake of brevity in this paper, only the two sample problems will be included to illustrate the use of PROBGEN. It should be mentioned, however, that PROBGEN can also employ library functions (e.g., LOG, SIN, etc.), and that it can accommodate up to a maximum of twenty variables within a given problem. Also, although the two examples used in this paper were both chemistry problems, it should be obvious that PROBGEN may be applied in other disciplines as well. Any numerical problem which can be systematically generalized into an answer equation(s) may be processed via PROBGEN, irrespective of whether the topic is physics, biology, or economics.

In terms of limitations, it is equally obvious that PROBGEN is applicable only to the generation of quantitative problems. It cannot be used to create theory or discussion questions, and thus PROBGEN cannot produce as wide a range of test types as can a CATC-R system. Also, in comparison to the very elaborate CATC-G system of Johnson, PROBGEN cannot generate problems with an accompanying set of multiple choice answers, including plausible distractors. On the positive side though, PROBGEN can be employed to create new problems by a faculty member who is a non-programmer whereas additions to the Johnson CATC-G system require considerable programming expertise. Thus, PROBGEN not only generates individualized, non-identical problem sets or tests for student use, but it permits the faculty member to easily and quickly (usually 15-20 minutes usage time) individualize the style of the problem according to his (her) pedagogical preferences.

On the subject of pedagogy, no attempt has been made in this paper to justify the use of individualized assignments or tests, nor to describe the proper logistics for their most effective use. Instead, the focus has been on the design and use of PROBGEN, and it is left for the reader to decide whether he (she) wishes to make use of such materials in his (her) teaching.

In conclusion, PROBGEN is still undergoing some modification; however, the present FORTRAN version is available upon request (source program on magnetic tape, plus listing) and its transportability from our DEC 10 system to other computing hardware should be reasonably easy. A concerted effort has been made to use only ANSI standard FORTRAN so as to make PROBGEN as transportable as possible. In addition to acquiring PROBGEN for direct implementation on other machines, a demonstration version of the program is available for remote access and experimental use by interested individuals willing to pay their own communication costs (for more information write to

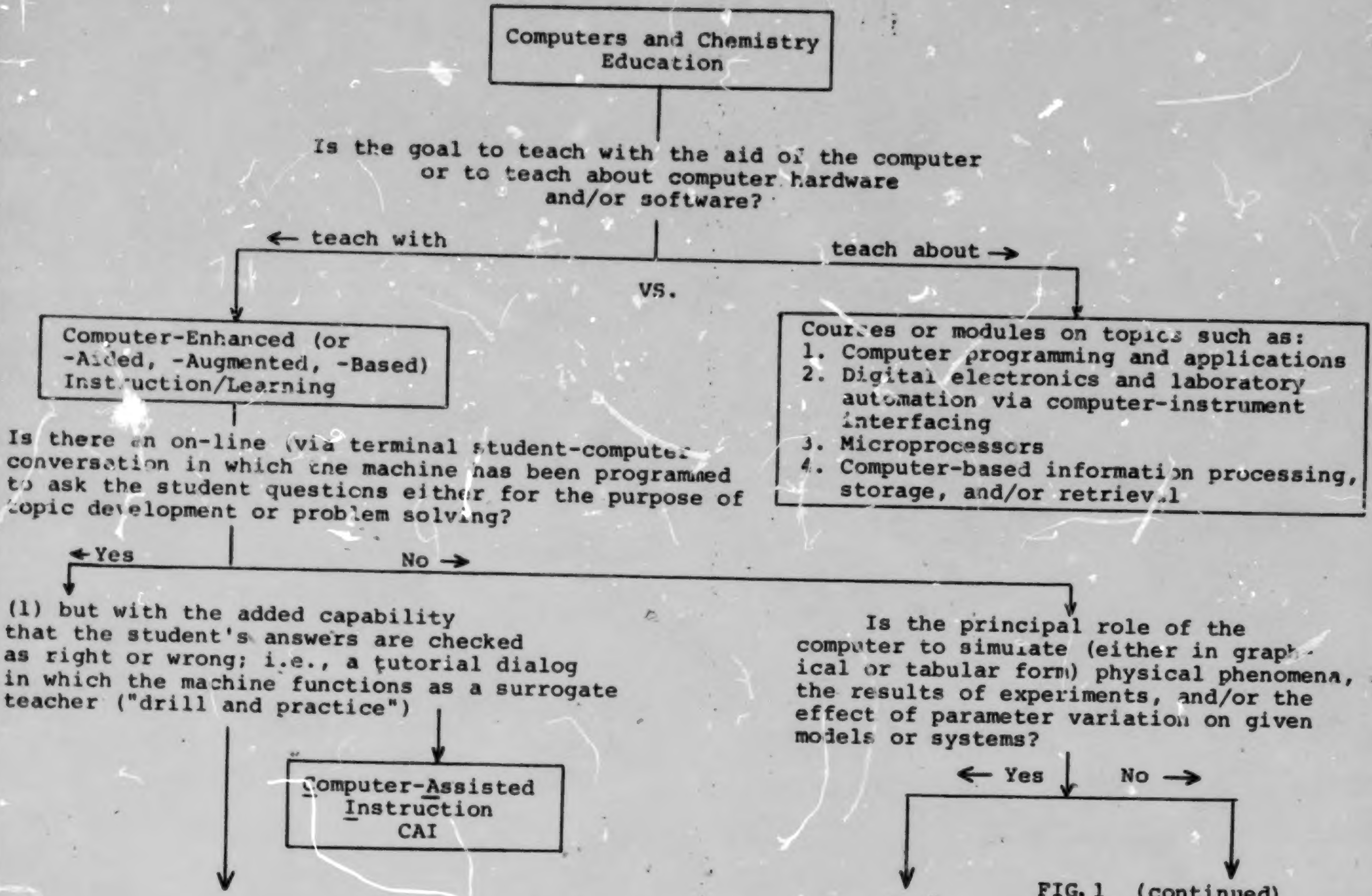
R. W. Collins).

REFERENCES

1. Computer-Assisted Test Construction, G. Lippey, Editor, Educational Technology Publications, Englewood Cliffs, N.J. 07632 (1974).
2. "Computer-Generated Repeatable Tests in Chemistry", J.W. Moore, Proceedings of the Conference on Computers in Chemical Education and Research, Northern Illinois University (1971).
3. "Repeatable Tests", F. Prosser, Educational Technology, 13 (3), 34 (1973).
4. "The SOCRATES CATC System at the California State Universities and Colleges", W.V. Willis and O. Seely, paper presented at the First Chemical Congress of the North American Continent, Mexico City, Mexico (1975).
5. For information/participation, write to Dr. W. Van Willis, Department of Chemistry, California State University - Fullerton, Fullerton, CA 92634.
6. "Pittsburgh's Computer-Generated Repeatable Exam System", K.J. Johnson, Proceedings of the 1973 Conference on Computers in the Undergraduate Curricula, p. 199 (1973).

Stoichiometry Problem		Gaseous Equilibrium Problem
Limiting reagent problem based on the hypothetical reaction $aA + bB \quad cC + dD$	General Concept of the Problem	Equilibrium constant problem based on the hypothetical gaseous system $aA(g) \quad cC(g) + dD(g)$
the balancing coefficients a, b, c, d the molecular weights of A, B, D the mass of A and B, the yield	Given (or to be assigned by the program)	the balancing coefficients a, c, d initial moles of A (A_i) equilibrium moles of A (A_e) volume of the container (V)
the mass of D produced in the reaction	To be calculated by the student	the value of the equilibrium constant K
(i) if $\frac{mass_A}{a \cdot MW_A} < \frac{mass_B}{b \cdot MW_B}$ $mass_D = \frac{(mass_A)\ (d)\ (MW_D)}{(a)\ (MW_A)}$ (ii) if $\frac{mass_B}{b \cdot MW_B} < \frac{mass_A}{a \cdot MW_A}$ $mass_D = \frac{(mass_B)\ (d)\ MW_D)}{(b)\ (MW_B)}$	Answer equation(s) and the special condition for each (if any)	$K = \frac{\left[\frac{c}{a}\left(\frac{A_i - A_e}{V}\right)\right]^c \left[\frac{d}{a}\left(\frac{A_i - A_e}{V}\right)\right]^d}{\left[\frac{A_e}{V}\right]^a}$ $A_i > A_e$ $a \neq c + d$

TABLE 1 Off-line Organization of Two Problems Prior to Interaction with PROBGEN



Computers and Chemistry Education

Is the goal to teach with the aid of the computer or to teach about computer hardware and/or software?

← teach with

vs.

teach about →

Computer-Enhanced (or -Aided, -Augmented, -Based) Instruction/Learning

Courses or modules on topics such as:
1. Computer programming and applications
2. Digital electronics and laboratory automation via computer-instrument interfacing
3. Microprocessors
4. Computer-based information processing, storage, and/or retrieval

Is there an on-line (via terminal student-computer conversation in which the machine has been programmed to ask the student questions either for the purpose of topic development or problem solving?

←Yes

No →

(1) but with the added capability that the student's answers are checked as right or wrong; i.e., a tutorial dialog in which the machine functions as a surrogate teacher ("drill and practice")

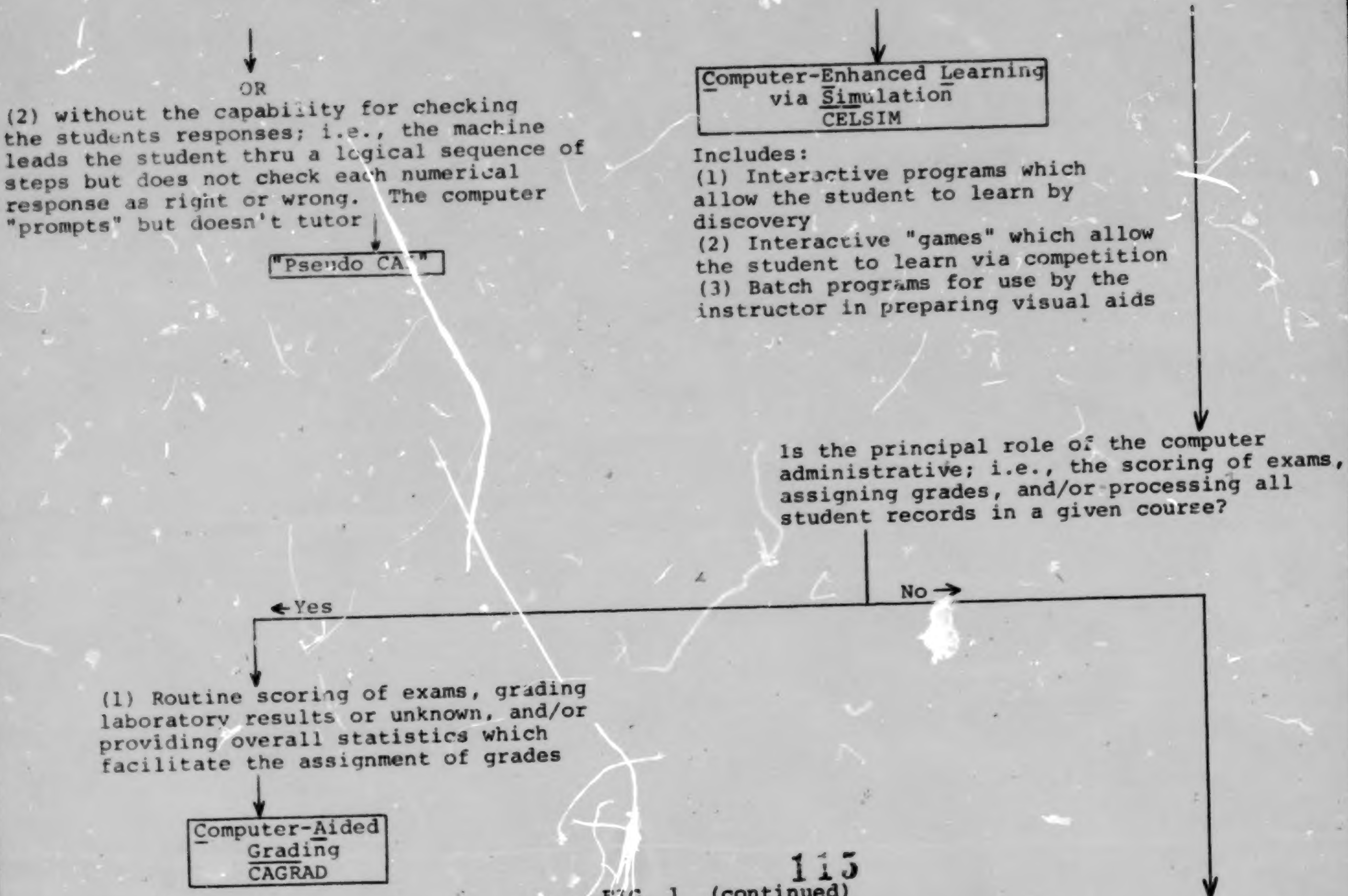
Computer-Assisted Instruction CAI

Is the principal role of the computer to simulate (either in graphical or tabular form) physical phenomena, the results of experiments, and/or the effect of parameter variation on given models or systems?

← Yes

No →

FIG. 1 (continued)



OR

(2) without the capability for checking the students responses; i.e., the machine leads the student thru a logical sequence of steps but does not check each numerical response as right or wrong. The computer "prompts" but doesn't tutor

"Pseudo CAI"

Computer-Enhanced Learning via Simulation
CELSIM

Includes:

(1) Interactive programs which allow the student to learn by discovery

(2) Interactive "games" which allow the student to learn via competition

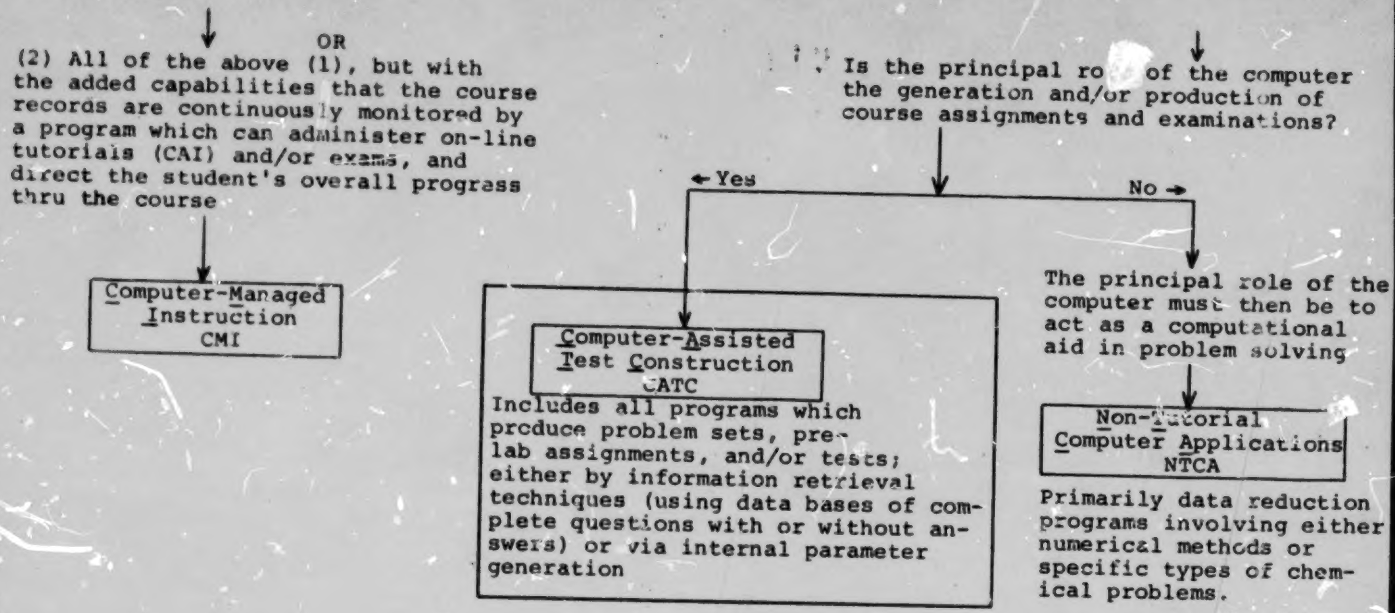
(3) Batch programs for use by the instructor in preparing visual aids

Is the principal role of the computer administrative; i.e., the scoring of exams, assigning grades, and/or processing all student records in a given course?

(1) Routine scoring of exams, grading laboratory results or unknown, and/or providing overall statistics which facilitate the assignment of grades

Computer-Aided Grading
CAGRAD

FIG. 1 (continued)



OR

(2) All of the above (1), but with the added capabilities that the course records are continuously monitored by a program which can administer on-line tutorials (CAI) and/or exams, and direct the student's overall progress thru the course

Computer-Managed Instruction CMI

Is the principal role of the computer the generation and/or production of course assignments and examinations?

Yes

No

Computer-Assisted Test Construction CATC

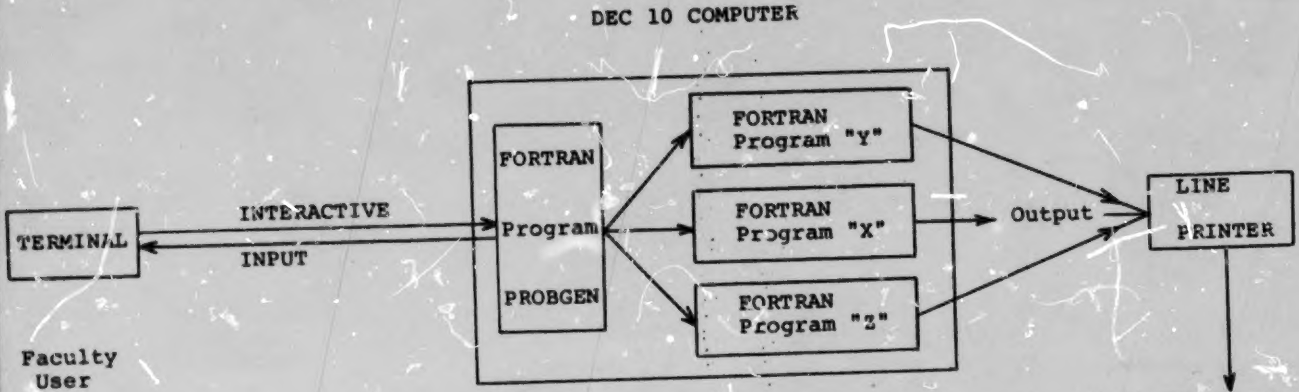
Includes all programs which produce problem sets, pre-lab assignments, and/or tests; either by information retrieval techniques (using data bases of complete questions with or without answers) or via internal parameter generation

The principal role of the computer must then be to act as a computational aid in problem solving

Non-Tutorial Computer Applications NTCA

Primarily data reduction programs involving either numerical methods or specific types of chemical problems.

FIGURE 1 Classification of the Various Roles of the Computer in Chemistry Education


DEC 10 COMPUTER
TERMINAL
INTERACTIVE
INPUT
Faculty User
FORTRAN Program PROBGEN
FORTRAN Program "Y"
FORTRAN Program "X"
FORTRAN Program "Z"
Output
LINE PRINTER

One or more programs are created by PROBGEN, one for each problem type X, Y, and/or Z as specified in the input.

Batch production of n tests or problem sets with accompanying answer key; each test consists in turn of one or more problems of each type, X, Y, and/or Z.

FIGURE 2 Outline of the Use and Operation of program PROBGEN at Eastern Michigan University

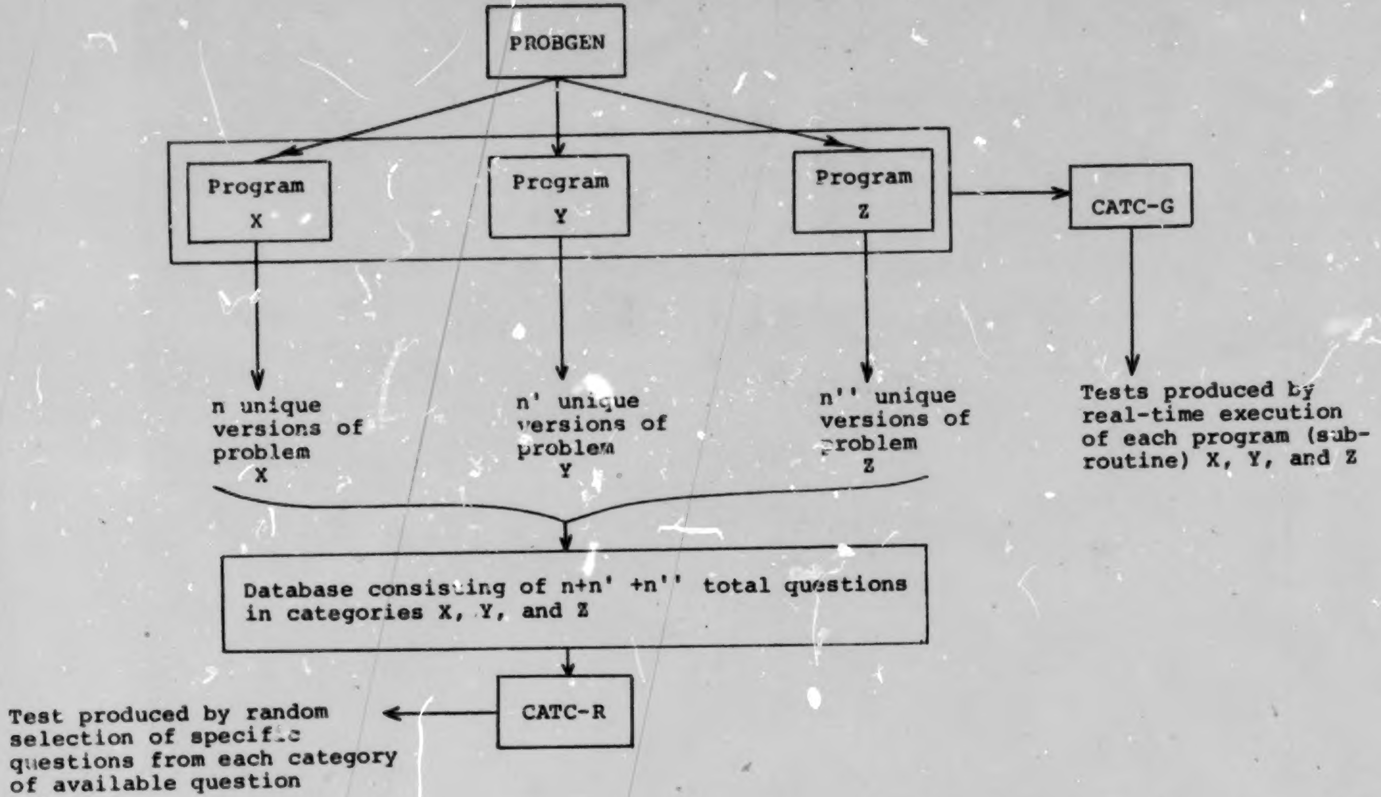

PROBGEN
Program X
Program Y
Program Z
CATC-G
n unique versions of problem X
n' unique versions of problem Y
n'' unique versions of problem Z
Tests produced by real-time execution of each program (subroutine) X, Y, and Z
Database consisting of n+n' +n'' total questions in categories X, Y, and Z
CATC-R
Test produced by random selection of specific questions from each category of available question

FIGURE 3 Use of PROBGEN in conjunction with either CATC-R or CATC-G Systems

```
..RUN PRBGEN

TYPE IN A HEADER FOR EACH PROBLEM-FORM. SHOULD INCLUDE:
DATE, INSTRUCTOR, ETC.  DO NOT TYPE BEYOND UPWARD ARROW (↑)!!!

TYPE:SAMPLE EXECUTION OF THE EMU PROGRAM PROBGEN

IS THIS THE RIGHT HEADER:

SAMPLE EXECUTION OF THE EMU PROGRAM PROBGEN
TYPE YES OR NO >YES
TYPE IN A HEADER -- WILL BE IN LARGE LETTERS
DO NOT TYPE BEYOND THE UPWARD ARROW (↑)

TYPE:EMU CHEM DEPT

IS THIS THE RIGHT LARGE HEADER:
EMU CHEM DEPT
TYPE YES OR NO >YES

HOW MANY PROBLEM FORMS OR SETS DO YOU WANT,
IN OTHER WORDS,HOW MANY STUDENTS IN YOUR CLASS>3

HOW MANY PROBLEMS OR QUESTIONS ARE TO BE PUT ON
EACH FORM>4
HOW MANY EQUATIONS USED IN THIS PROBLEM:2
INPUT CONDITION FOR EQUATION  1:(GRAMA/(NA*AMW)).LT.(GRAMB/(NB*BMW))

INPUT EQUATION  1:(GRAMA*ND*DMW*NB*NC*BMW*GRAMB)/(AMW*NA*NB*NC*BMW*GRAMB)
INPUT CONDITION FOR EQUATION  2:(GRAMB/(NB*BMW)).LT.(GRAMA/(NA*AMW))

INPUT EQUATION  2:(GRAMB*ND*DMW*NA*NC*AMW*GRAMA)/(BMW*NB*NA*NC*AMW*GRAMA)

IS GRAMA A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>75.0

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>15.0

IS ND    A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>8

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>2

IS DMW   A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>200.0

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>75.0

IS NB    A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>8

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>2
```

Three problem sets or tests consisting of four problems each

(Stoichiometry) Start of the first problem

Variables are called, and limits are specified, in the order in which they appear in the answer equation(s)

FIGURE 4(a) Sample User Interaction with PROBGEN

```
IS NC    A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>8

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>2

IS BMW   A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>150.0

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>75.0

IS GRAMB A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>75.0

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>15.0

IS AMW   A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>150.0

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>75.0

IS NA    A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>8

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>2
ARE THERE ANY RESTRICTIONS IN THE RANCOM INTERVALS:NO
HOW MANY LINES IN YOUR TEXT OF THE QUESTION:7

TYPE IN THE TEXT OF YOUR QUESTION MAKING SURE YOU PUT A "@"
SYMBOL IS IN FRONT OF YOUR VARIABLE-NAME
IN ITS PROPER PLACE IN THE TEXT AND ALSO
MAKE SURE YOU TEXT DOESN'T GO BEYOND FOUR LINES
(ADD BLANKS WHEN NECCESSARY) -----
DO NOT TYPE BEYOND UPWARD ARROW (↑)--
```

Directions for composing the text of the first problem

```
TYPE:ASSUMING A 100 PERCENT YIELD FOR THE REACTION
GIVEN BELOW, CALCULATE THE MASS (GRAMS) OF D (MW=@DMW )
PRODUCED WHEN @GRAMA GRAMS OF A (MW=@AMW ) REACTS WITH
@GRAMB GRAMS OF B (MW=@BMW ).

   @NA A    +    @NB B    --------->  @NC C    +    @ND D
```

Text of the first problem

```
IS YOUR TEXT OK?  HERE IT IS:
ASSUMING A 100 PERCENT YIELD FOR THE REACTION
GIVEN BELOW, CALCULATE THE MASS (GRAMS) OF D (MW=@DMW )
PRODUCED WHEN @GRAMA GRAMS OF A (MW=@AMW ) REACTS WITH
@GRAMB GRAMS OF B (MW=@BMW ).

   @NA A    +    @NB B    --------->  @NC C    +    @ND D
```

Repeat of the text - check for typographical errors

```
IS THE TEXT OK, CHECK YOUR VARIABLES
TYPE YES OR NO >YES
```

(Stoichiometry) Start of the second problem

```
HOW MANY EQUATIONS USED IN THIS PROBLEM:1
```

FIGURE 4(b) Sample User Interaction with PROBGEN (continued)

```
TYPE IN THE EQUATION THAT PRODUCES THE ANSWER
TO THE PROBLEM--REMEMBER INTEGER VARIABLES START WITH
ONE OF THE FOLLOWING! I,J,K,L,M,N AND REAL VARIALBES--ANY THING ELSE.
IF YOU WANT TO USE A FUNCTION-- TYPE THE "@" IN FRONT OF IT.
FOR EXAMPLE! @SIN OR @COS     Second problem will be the same type as the
TYPE THE EQUATION:SAME        first problem

HOW MANY EQUATIONS USED IN THIS PROBLEM!1
                                               Start of the third problem
TYPE IN THE EQUATION THAT PRODUCES THE ANSWER    (Equilibrium)
TO THE PROBLEM--REMEMBER INTEGER VARIABLES START WITH
ONE OF THE FOLLOWING! I,J,K,L,M,N AND REAL VARIALBES--ANY THING ELSE.
IF YOU WANT TO USE A FUNCTION-- TYPE THE "@" IN FRONT OF IT.
FOR EXAMPLE! @SIN OR @COS
TYPE THE EQUATION:((((NC*(AX-AA))/(NA*V))**NC)*(((ND*(AX-AA))/(NA*V))**ND))

IS NC     A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>7

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>2

IS AX     A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>10.0

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>0.10

IS AA     A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>10.0

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>0.10

IS NA     A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>7

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>2

IS V      A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>15.0

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>1.5

IS ND     A RANDOM OR CONSTANT - TYPE R OR C >R

TYPE IN THE UPPER LIMIT OF THE RANDOM INTERVAL>7

TYPE IN THE LOWER LIMIT OF THE RANDOM INTERVAL>2           Restrictions as
ARE THERE ANY RESTRICTIONS IN THE RANDOM INTERVALS!YES    cited in Table 1
WHAT IS THE VARIABLE THAT NEEDS TO BE
RESTRICTED WITHIN THE RANDOM INTERVAL!NA
TYPE ".LT.",".GT.",".LE.",".GE.",".EQ.",".NE." IN
APPROPRIATE RELATION TO THE CONDITIONAL STATEMENT!.EQ.
ENTER IN THE ALEBRAIC CONDITIONAL STATEMENT!(NC+ND)
ARE THERE ANY MORE RESTRICTIONS!YES
WHAT IS THE VARIABLE THAT NEEDS TO BE
```

FIGURE (4c) Sample User Interaction with PROBGEN (continued)

```
RESTRICTED WITHIN THE RANDOM INTERVAL:AA
TYPE ".LT.",".GT.",".LE.",".GE.",".EQ.",".NE." IN
APPROPRIATE RELATION TO THE CONDITIONAL STATEMENT:.GE.
ENTER IN THE ALEBRAIC CONDITIONAL STATEMENT:AX
ARE THERE ANY MORE RESTRICTIONS:NO
HOW MANY LINES IN YOUR TEXT OF THE QUESTION:8

TYPE IN THE TEXT OF YOUR QUESTION MAKING SURE YOU PUT A "@"
SYMBOL IS IN FRONT OF YOUR VARIABLE-NAME
IN ITS PROPER PLACE IN THE TEXT AND ALSO
MAKE SURE YOU TEXT DOESN'T GO BEYOND FOUR LINES
(ADD BLANKS WHEN NECCESSARY) -----
DO NOT TYPE BEYOND UPWARD ARROW (↑)--

TYPE:EXACTLY @AX MOLES OF A(G) WERE INJECTED INTO A
@V LITER FLASK AT 298 DEGREES KELVIN, UPON DECOMPOSI-
TION ACCORDING TO THE REACTION BELOW, AN EQUILIBRIUM
WAS ESTABLISHED AND @AA MOLES OF A(G) REMAINED IN THE
FLASK AT THE SAME TEMPERATURE.  CALCULATE THE EQUIL-
BRIUM CONSTANT, K, FOR THIS REACTION AT 298 DEGREES.

@  @NA A(G)   <------>   @NC C(G)   +   @ND D(G)
```

Text of the third problem

```
IS YOUR TEXT OK?  HERE IT IS:
EXACTLY @AX MOLES OF A(G) WERE INJECTED INTO A
@V LITER FLASK AT 298 DEGREES KELVIN, UPON DECOMPOSI-
TION ACCORDING TO THE REACTION BELOW, AN EQUILIBRIUM
WAS ESTABLISHED AND @AA MOLES OF A(G) REMAINED IN THE
FLASK AT THE SAME TEMPERATURE.  CALCULATE THE EQUIL-
BRIUM CONSTANT, K, FOR THIS REACTION AT 298 DEGREES.

@  @NA A(G)   <------>   @NC C(G)   +   @ND D(G)

IS THE TEXT OK, CHECK YOUR VARIABLES
TYPE YES OR NO >YES
```

Start of the fourth problem (Equilibrium)

```
HOW MANY EQUATINS USED IN THIS PROBLEM:1

TYPE IN THE EQUATION THAT PRODUCES THE ANSWER
TO THE PROBLEM--REMEMBER INTEGER VARIABLES START WITH
ONE OF THE FOLLOWING: I,J,K,L,M,N AND REAL VARIABLES--ANY THING ELSE.
IF YOU WANT TO USE A FUNCTION-- TYPE THE "@" IN FRONT OF IT.
FOR EXAMPLE: @SIN OR @COS
TYPE THE EQUATION:SAME
FORTRAN:  QUEST.F4
LINK:  LOADING
```

Fourth problem will be the same type as the third problem

FIGURE 4(d) Sample User Interaction with PROBGEN (continued)

```
EEEEE M   M U   U      CCC  H   H EEEEE M   M       DDDD  EEE
E     MM MM U   U     C   C H   H E     MM MM       D   D E
E     M M M U   U     C     H   H E     M M M       D   D E
EEEE  M   M U   U     C     HHHHH EEEE  M   M       D   D EEE
E     M   M U   U     C     H   H E     M   M       D   D E
E     M   M U   U     C   C H   H E     M   M       D   D E
EEEEE M   M  UUU       CCC  H   H EEEEE M   M       DDDD  EEE
```

```
1.  ASSUMING A 100 PERCENT YIELD FOR THE REACTION
    GIVEN BELOW, CALCULATE THE MASS (GRAMS) OF D (MW=109.81 )
    PRODUCED WHEN 18.40 GRAMS OF A (MW=149.89 ) REACTS WITH
    66.71 GRAMS OF B (MW=146.70 ).

        6 A   +    6 B  ------->   3 C   +    8 D
```

First two problems, same type, but with different numbers and equations

```
2.  ASSUMING A 100 PERCENT YIELD FOR THE REACTION
    GIVEN BELOW, CALCULATE THE MASS (GRAMS) OF D (MW=172.05 )
    PRODUCED WHEN 69.03 GRAMS OF A (MW=127.07 ) REACTS WITH
    42.06 GRAMS OF B (MW=146.39 ).

        3 A   +    7 B  ------->   2 C   +    6 D
```

Problems 3 and 4 (next page), same type but with different numbers and equations

```
3.  EXACTLY  2.18 MOLES OF A(G) WERE INJECTED INTO A
     5.58 LITER FLASK AT 298 DEGREES KELVIN. UPON DECOMPOSI-
    TION ACCORDING TO THE REACTION BELOW, AN EQUILIBRIUM
    WAS ESTABLISHED AND  1.10 MOLES OF A(G) REMAINED IN THE
    FLASK AT THE SAME TEMPERATURE.  CALCULATE THE EQUIL-
    IBRIUM CONSTANT, K, FOR THIS REACTION AT 298 DEGREES.

        7 A(G)   <------->    7 C(G)   +    4 D(G)
```

FIGURE 5(a) Sample Problem Set/Test as created by PROBGEN

```
EXACTLY  7.74 MOLES OF A(G) WERE INJECTED INTO A
 2.42 LITER FLASK AT 298 DEGREES KELVIN. UPON DECCMPOSI-
TION ACCORDING TO THE REACTION BELOW, AN EQUILIBRIUM
WAS ESTABLISHED AND  1.82 MOLES CF A(G) REMAINED IN THE
FLASK AT THE SAME TEMPERATURE.  CALCULATE THE EQUIL-
IBRIUM CONSTANT, K, FOR THIS REACTICN AT 298 DEGREES.

    3 A(G)   <------->    2 C(G)   +    3 D(G)



             ***ANSWERS TO ALL PROBLEMS***


        PROB#  ANSWERS  VARIABLES IN TEXT (10TH TIME)

FORM#  1
           1      31.38295
           2      19.76344
           3     0.1971930E-01
           4     0.5926250
FORM#  2
           1      32.37072
           2      83.28945
           3     0.5516171E-01
           4     0.2139728E-09
FORM#  3
           1      13.47920
           2      41.79955
           3     0.3254437E-04
           4      91.71787
```

Answer key is printed
at the end of the last
test/assignment produced

FIGURE 5(b) Sample Problem Set/Test as Created by PROBGEN (continued)

GENERATIVE CAI IN ARISTOTELIAN LOGIC

Dr. Walter Maner

Old Dominion University

1.0 INTRODUCTION

According to some researchers, the future of ordinary CAI, with its canned questions and repertoire of canned answers, grows dimmer by the day. They would support the use of the tedious frame-by-frame approach for only a few more years while educational technicians ready the more powerful generative and simulation techniques.[1]

It is not hard to see why. Once a GCAI program has been designed, it is capable of furnishing an inexhaustible supply of distinct problems (and solutions) for the student. By contrast, in a frame-oriented CAI system, the lesson designer typically programs each task as a separate case. Like ordinary programming but unlike conventional CAI, GCAI uses algorithms to manipulate structured data in order to produce output--in this case, a series of learning tasks for the student. There is no GCAI 'language' in the tradition of GNOSIS, TUTOR, MENTOR, SCHOLAR-TEACH, COURSEWRITER, PILOT, and so on. Instead of constructing frames, the GCAI author uses some general-purpose, high-level programming language such as SIMULA, PL/I or SAIL to build the data structures and algorithms which will (a) generate problems, (b) solve these problems, (c) present these problems (unsolved) to the student, (d) manage the dialogue between student and program (much as in conventional CAI), and (e) evaluate the student's problem-solving efforts (again, much as in conventional CAI).

The SPECIFIC nature of the learning tasks set for the student by GCAI cannot be anticipated, of course, and this unknown greatly complicates the problem of recognizing correct responses and providing remedial feedback. To render this problem more nearly manageable, some GCAI authors severely limit the range of tasks generated by the program. In fact, a frequent criticism of GCAI is that it is inherently limited in the difficulty of the problems it can handle. 'Only problems of an algorithmic nature can be easily

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

programmed as course concepts,' observes Hickey. 'Problems requiring insight on the part of the computer would be immensely difficult to program and would lengthen computer response time.'[2] The program described below refutes some of this pessimistic theorizing, and it shows how heuristic rules can be used in GCAI to produce answers to unanticipated problems which have no clear algorithmic solutions.

2.0 DESCRIPTION OF THE VENN PROGRAM

VENN[3] is a program which, through generative routines, is able to provide virtually inexhaustible resources for computer-assisted practice in traditional syllogistic reasoning. Such reasoning involves the inference of one proposition (the conclusion of the syllogism) from two other propositions (the major and minor premises). Each premise is constructed in such a way that it has one category term in common with the conclusion and another in common with the other premise. Each proposition contains exactly two terms, linked by 'are' or 'are not,' and the first of these terms is preceded by 'all,' 'no,' or 'some.' For as long as the student wishes to work, VENN will create interesting and original syllogisms for him to analyze in detail, e.g.,

```
Major Premise:  No READY ROMANCERS are BACKSEAT DRIVERS.
Minor Premise:  Some BACKSEAT DRIVERS are COMPULSIVE LIARS.
-----------------------------------------------------------
Conclusion:     No COMPULSIVE LIARS are READY ROMANCERS.
```

The effect is the same as if the program were picking them at random from a database containing some twenty-seven million different syllogistic arguments.

In the course of generating the syllogism, VENN analyzes the argument internally to determine (among other things) its mood, figure, validity, diagrammatic representation, and--if invalid--the exact reason(s) for its invalidity. It then queries the student on each of these points, providing brief help messages and default responses to simplify his interaction with the program.

Twice during the execution of the program, a summary of correct answers to previous questions is displayed, but this normally occurs only after the student has already discovered on his own which answers are indeed correct. For example, after the student has demonstrated that he knows what kind of mark belongs in each sector of the Venn diagram ('\', '/', 'X', '*', '?' or 'BLANKS'), the program proceeds to mark the diagram for him:

```
  ------------------       MAJOR CLASS:  READY ROMANCERS
M !      !  !      ! M     MINOR CLASS:  COMPULSIVE LIARS
I !      !  !      ! A     MIDDLE CLASS: BACKSEAT DRIVERS
N !   ----------   ! J     MOOD-FIGURE:  EIE-4
O !   !**!//!//!   ! O     VIEWPOINT:    Boolean
R !   !**!//!//!   ! R
  ------------------
      !        !
      !        !
      ----------

        MIDDLE
```

Next it asks him whether or not the syllogism is valid and, after an acceptable answer, it pauses to display all the valid inferences (if any) which could have been made from the same two premises, e.g., Conclusion: Some COMPULSIVE LIARS are not READY ROMANCERS.

During this pause the student can verify for himself that those listed are the ONLY possible valid inferences by referring again to the Venn diagram. If the syllogism happens to be invalid,[4] the student is quizzed about the reason(s) for its invalidity. Ultimately, a complete data summary is output to the student:

```
  ---------------------         MAJOR CLASS:    READY ROMANCERS
M !       !  !        ! M     MINOR CLASS:    COMPULSIVE LIARS
I !       !  !        ! A     MIDDLE CLASS: BACKSEAT DRIVERS
N !   -----------     ! J     MOOD-FIGURE:    EIE-4
O !    !**!//!//!     ! O     VIEWPOINT:      Boolean
R !    !**!//!//!     ! R     VALID:          No
  ---------------------         FALLACY(7):     Illicit minor
       !         !
       !         !
       -----------

         MIDDLE
```

Before the program recycles to create another syllogism, it tries to generate a plausible counterexample to the syllogism just analyzed. This involves a systematic replacement of the original terms of the syllogism by other terms in such a way that both premises become clearly TRUE while the conclusion becomes clearly FALSE:

```
No FIVE-YEAR-OLD BOYS are FORMER US PRESIDENTS.
Some FORMER US PRESIDENTS are LIVING THINGS.
----------------------------------------------------------
No LIVING THINGS are FIVE-YEAR-OLD BOYS.
```

The semantic heuristics built into the program (see discussion below, section 3.2) succeed in providing a counterexample for approximately two-thirds of all invalid syllogisms.

At his option, the student may skip the question-and-answer routines and receive the final data summary at once. Since the nature of the syllogism to be generated can be specified with any degree of completeness, this skipping option allows the student to use VENN for something as specific as checking homework exercises or for something as general as exploratory learning. He may, for example, be curious to see how a syllogism of the same mood would fare if it were cast in another figure or if it were analyzed under the other interpretation (e.g., Aristotelian instead of Boolean). Or he may want to try replacing elements of a defective syllogism one at a time until it is rendered valid. Perhaps the teacher will challenge him to get VENN's help in constructing a series of one-fallacy syllogisms, each one exemplifying a different fallacy. The possibilities are many, and VENN will follow cheerfully wherever curiosity may lead.

While several programs currently exist for drawing Venn diagrams,[5] VENN is unique in several ways. It is the first generative program developed for teaching syllogistic reasoning. It is, moreover, the first program of its kind which works under either the Boolean or Aristotelian interpretations. And it is the first program to employ heuristic rules for constructing counterexamples. The author also believes it may be the first program which permits the user to control the nature of the syllogism displayed to any degree he wishes, from an extreme of no control on the one hand to an extreme of complete control on the other. It may also be the first program to query the student so thoroughly on facts pertinent to developing an understanding of the syllogism under consideration.

3.0 COMMENTS ON THE PROGRAM

3.1 How VENN Determines Validity

Internally, VENN determines validity in three different ways: (1) by comparing the form of the syllogism generated with a list of valid forms, (2) by "reading" the Venn diagram and (3) by a process of fallacy elimination. These mechanisms are independent, but not wastefully so, since they permit the program to monitor itself for internal errors.

Of these, "(3)" is the one most like the procedures human beings would use in the same circumstances. To see how it works, let us consider the premises of the syllogism used in the previous examples. After the Major Premise has been generated, VENN knows that fallacies #2 (undistributed middle term), #5 (negative conclusion from all positive premises) and #6 (illicit minor) can be eliminated from consideration. Then, after generation of the Minor Premise, VENN realizes that fallacies #1 (premises all negative), #4 (affirmative conclusion from a negative Minor Premise) and #8 (the existential fallacy) are no longer possible. This leaves #3 (affirmative conclusion from a negative Major Premise), and #7 (illicit minor) still threatening. The generation of a negative conclusion eliminates #3, however, leaving #7 as the sole mistake in reasoning.

By a process of addition VENN, in a manner somewhat analogous to this, determines which marks belong in which sectors of the Venn diagram. This procedure is, however, considerably complicated by (1) the observance of a precedence relation among the marks and (2) the need to change the '?' to a '*' when it becomes certain that the sector has a member.

3.2 How VENN Uses Heuristic Rules To Create Counterexamples

Since counterexamples must contain plausible true and false statements about the real world, any computer program which attempts to construct them must be told enough about that universe of discourse to be able to generate such statements. What little VENN knows is exhaustively summarized in the following semantic presumptions or rules-of-thumb:

1. VENN presumes (a) that some arbitrarily selected member of the following set 'B' of set labels

   LIVING THINGS
   CELLULAR ORGANISMS
   MORTAL THINGS
   CHROMOSOMAL CREATURES
   BIOLOGICAL ENTITIES
   BUNDLES OF PROTOPLASM
   FORMS OF LIFE

   should be used in a proposition when the objective is to make reference to a set having a large number of members and (b) that two arbitrarily selected members of 'B' should be used in a proposition when the objective is to make reference to coextensive sets (i.e., to sets whose symmetric difference is the null set).

2. VENN presumes (a) that some arbitrarily selected member of the following set 'L' of set labels

   WORLD BANK DIRECTORS
   MULTIBILLIONAIRES
   NOBEL LAUREATES
   OLYMPIC GOLD MEDALISTS
   SOVIET CHESS CHAMPIONS
   LUNAR EXPLORERS
   FORMER US PRESIDENTS
   FIVE-YEAR-OLD BOYS
   FIVE-YEAR-OLD GIRLS
   AFRICAN HEADS OF STATE

   should be used in a proposition when the objective is to make reference to a set having very few members and (b) that two non-identical but otherwise arbitrarily selected members of 'L' should be used in a proposition when the objective is to make reference to mutually exclusive sets (i.e., to sets whose intersection is the null set).

3. VENN presumes, finally, that some arbitrarily selected member of the following set 'M' of set labels

   HOMOSAPIENS
   RATIONAL BEINGS
   PROMISE MAKERS
   CONSCIOUS BEINGS
   HUMAN BEINGS
   PERSONS
   POLITICAL ANIMALS
   FEATHERLESS BIPEDS

   should be used in propositions when the objective is to make reference to some set (a) which is a proper subset of some arbitrary B-labeled set and (b) which has some arbitrary L-labeled set as a proper subset.

A list of objectives of the kind mentioned above is established for each proposition at the time the proposition is generated. For example, if the premise is of type 'A' (i.e., "All S is P"), then--in order to make it true--VENN's initial objective will be to replace both 'S' and 'P' with B-type labels. Such terms will likely be coextensive, thereby rendering the proposition true. If that can't be done consistently given the objectives for the OTHER propositions, then VENN tries replacing 'S' with an M-type label and 'P' with a B-type label. Since M-labeled sets are likely to be subsets of B-labeled sets, this kind of substitution will also tend to make the original proposition true. If this also fails, VENN will (for similar reasons) try substituting an L-type label for 'S', then first a B-type and finally an M-type label for 'P'.

The following is a summary table of the sequence of objectives VENN sets for replacing terms in order to render propositions of type 'A', 'E', 'I' or 'O' either true (when they are premises) or false (when they are conclusions):[6]

True 'A' False 'O'		True 'E' False 'I'		True 'I' False 'E'		True 'O' False 'A'	
S	P	S	P	S	P	S	P
B	B			B	B		
				B	M	B	M
				B	L	B	L
M	B			M	B		
				M	M		
				M	L	M	L
L	B			L	B		
L	M			L	M		
		L	L			L	L

Mnemonic: (B)ig, (M)edium, (L)ittle

In practice, VENN first tries to make the conclusion false then, as a subgoal to that goal, tries to make the Minor Premise true; finally, as a subgoal to that subgoal, VENN tries to make the Major Premise true. If these goals can be achieved consistently within the framework imposed by the heuristic rules and the original syllogism, VENN will output a counterexample to the student. Otherwise, VENN will merely report how close he came to success before giving up.

3.3 Design Compromises

3.3.1 Squares, Not Circles -

In the interest of making VENN's graphic output fast and terminal independent, overlapping squares were substituted for the more traditional overlapping circles. Since virtually all introductory logic textbooks use Venn or Euler circles, however, students commonly need four or five practice runs to adjust perceptually to VENN's squares.

3.3.2 Marking Conventions -

It is common to use the shading characters '/' and '\' to mark those Venn circle sectors which are declared empty by the premises. To identify the NONempty sectors, either (a) an 'X' or (b) an '*' and a bar of variable length and orientation is often used.

Since the '/' and the '\' occasionally intersect, howver, and since VENN asks the student about the FINISHED diagram, it was necessary to reject (a) and use 'X' strictly as a third shading character. Option (b), likewise, was rejected because of the considerable inefficiency involved in computing the length, orientation and point of insertion for the bar.

The marking conventions actually used by VENN are simple and straightforward:

'/'   = this sector empty by Major Premise
'\'   = this sector empty by Minor Premise
'X'   = this sector empty by BOTH the above
'?'   = this OR adjacent sector must have a member
'*'   = this sector must have a member
'BLANKS' = none of the above

One might think that the '?' would be even more ambiguous than required since it is not visually paired with a '?' in a neighboring sector. In practice, however, this does not cause any difficulty.

3.3.3 Minimal Existential Assumptions -

Under the Aristotelian (or existential) interpretation, VENN assumes that the subject class of each premise has at least one member, even if the premises are universal (i.e., type 'A' or 'E'). In many systems of syllogistic logic, however, it is assumed that EACH class mentioned--three in all--must be nonempty. VENN's somewhat stingier outlook is partly motivated by a desire to keep the diagramming task simple for the student. But the main motivation lies in a desire to have a definition of 'existential import' for three-line syllogistic inferences which is identical to the one normally used for two-line or 'immediate' inferences. Obviously, there are advantages and disadvantages to this decision.

The chief advantage, beyond that involved in not 'multiplying entities beyond necessity,' is that it becomes possible to use a series of immediate inferences to prove syllogisms valid or invalid. The method, which has been described by Terrell,[7] involves a series of appeals to some one previous line of proof and some principle of immediate inference (i.e., contradiction, conversion, obversion, contraposition or 'complement conversion', and limitation) plus an occasional appeal to some two previous lines of proof and the 'Principle of the Syllogism':

> In the first figure, when the major premise is universal and the minor premise is affirmative, a conclusion can be drawn that has the quantity of the minor premise and the quality of the major premise.[8]

A subsequent version of VENN will incorporate a proof checker for such deductions. The student will then have an opportunity to enter his proof one step at a time, always under VENN's watchful eye, and thereby gain experience which will be valuable later when he is asked to construct deductions in sentential logic.[9]

The disadvantage is that syllogisms of the form AEO-4, usually considered valid, turn out to be problematic under the stingier Aristotelian interpretation even though such arguments don't violate any of the traditional rules. The latin scholars would not have considered this much of a loss, however, and it is easy to modify the 'existential fallacy' rule to outlaw AEO-4.[10] At present, however, the program does consider AEO-4 to be valid even though it fails the second of the three tests used for determining validity (see section 3.1, above).

3.3.4 A Proliferation Of Rules -

VENN applies eight rules to check for fallacies in syllogistic reasoning. Several of these could be consolidated, of course, but this would not be appropriate given the many different ways textbook authors state 'the' rules.

4.0 TECHNICAL REMARKS

1. System Description.

VENN was developed on a DEC PDP-10 computer with 512K words of memory (36-bit words) under a KL 602A+VM monitor at Old Dominion University, Norfolk, Virginia.

2. Source Language.

VENN is written in SIMULA, a general purpose, high-level, ALGOL-like programming language comparable in power and features to PL/I, ALGOL68, ALGOLW or SAIL. Specific features worth noting include record-oriented dynamic memory allocation, efficient garbage collection, reference (pointer) structures, list structures, sets and queues, text- and character-handling procedures, sequential and direct access input/output, quasi-parallel processing via coroutines, and process- or event-oriented simulation capabilities.

Like PASCAL, SIMULA was designed to be not only a language but also a tool of thought in its own right which, by its very use, would steer the programmer toward the production of well-structured, highly readable and secure programs. SIMULA is especially well suited to programming methodologies which stress 'modular construction,' 'top-down development' or 'stepwise refinement.'[11] Despite its length (see below), VENN needs only three GOTOs.

3. Length.

VENN.SIM requires over 1600 lines of source code (including comments), occupies about 135 blocks of disk storage, and contains about 16000 DEC words. VENN.SAV occupies about 150 blocks of disk storage and contains about 18700 DEC words.

4. Core and CPU Requirement.

The execution of VENN.SAV requires a maximum of 47+12 pages (or about 30K) of DEC core. One cycle through the complete program with an invalid syllogism, accepting default answers wherever possible, typically requires about 1.3 seconds of CPU time. (SIMULA text-handling is extremely fast compared to other ALGOL-type languages.)

5. Other Software Required.

VENN.SAV, of course, requires the SIMULA run-time system. In addition, recompilation of VENN.SIM (e.g., after modification) would require the presence of essential compiler files, the SIMULA libraries 'SIMLIB' and 'LIBSIM' (plus their '.ATR' files) in the 'SYS:' area. Given a normal implementation of SIMULA, however, no difficulties in recompilation should be experienced.

6. Distribution.

VENN is in the public domain and is distributed worldwide by the DECUS Library, Digital Equipment Computer Users Society, 146 Main Street, Maynard, Massachusetts 01754.

7. Portability.

Although no particular attention has been given to making VENN portable to other major SIMULA installations (i.e., IBM 360/370, CDC 6000 and UNIVAC 1100), VENN does adhere closely to 'common base' SIMULA and should not be too difficult to implement on non-DEC SIMULA systems.[12] Version 2 will address portability problems directly.

FOOTNOTES

1. For a look at both sides of this issue, see A. E. Hickey, 'Research Guidelines for Computer-Assisted Instruction,' CONCEPT MONOGRAPHS (Newburyport, Mass.: A. E. Hickey, 1974). I continue to support both approaches. In addition to the generative program described here, for example, I have co-authored (with Jacob Palme) a new version of the CAI author language GNOSIS (actually an ALGOL preprocessor). Version 2 of GNOSIS, soon to be submitted to the DECUS Library, will be the most powerful test-and-branch, frame-oriented CAI software available on the PDP-10.

2. A. E. Hickey, COMPUTER-ASSISTED INSTRUCTION: A SUMMARY OF RESEARCH IN SELECTED AREAS (Newburyport, Mass.: Entelek, Inc., 1975), p. 39 (approx.).

3. Developed by the author. Named after the English logician John Venn (1834-1923), who developed diagrams of the type used in the program. As Venn knew, these diagrams were generalizations of algebraic processes which had already been introduced by Boole and illustrated in Jevons' 'logical alphabet.'

4. If the student imposes no restrictions on the generation process, the chance of his receiving a valid syllogism is about 1 in 11 under the Aristotelian interpretation or about 1 in 17 under the Boolean interpretation.

5. Dr. Richard Wright, Department of Philosophy, University of Toledo, has authored one of the better programs for mid-size computers: CATSYL.BAS

6. For no particular reason, VENN now prefers B-labels to M-labels to L-labels. The author is currently investigating whether any small advantage can be gained by resequencing VENN's path through these objectives.

One would think that a substantial improvement could be realized by having TWO 'M' classes, one for referring to coextensive sets (like 'B') and another (like 'L') for referring to exclusive sets. But M-labeled sets must each contain as subsets all L-labeled sets, hence could not have null intersections with each other.

7. D. B. Terrell, LOGIC: A MODERN INTRODUCTION TO DEDUCTIVE REASONING (New York: Holt, Rinehart and Winston, 1967), sections 4.8 ff.

8. Terrell, p. 94.

9. It probably would not be too difficult to add a GPS-like theorem PROVER to VENN for generating such proofs. Part or (finally) all of the program-generated proof could then be displayed should the student become stuck.

10. Instead of

(8) Both premises universal, yet conclusion is particular

VENN could simply pose

(8) Minor class nonempty by Conclusion but not by Minor Premise

as a true/false response item. Since AEO-4 is already diagram-invalid, the only other change required in the program would be the deletion of AEO-4 from the list of valid syllogisms.

11. See O.-J. Dahl, E. W. Dijkstra, and C. A. R. Hoare, STRUCTURED PROGRAMMING (New York: Academic Press, 1972). Part III is entirely devoted to SIMULA.

12. The SIMULA library contains programs which will perform nearly all of the source code translation required for interchange of the software between SIMULA installations. The author will be glad to try to prepare exportable versions upon request.

GENERATING ENGLISH PROGRAMS AT A SMALL COLLEGE

Gayle Byerly

Ursinus College

The computer arrived at Ursinus College, a small liberal arts school located in eastern Pennsylvania, in the fall of 1973. The faculty was introduced to the benefits of our Honeywell 635 computer and the Dartmouth time-sharing system (DTSS) by a salesman who assured us that the new technology would revolutionize our teaching. Most of my colleagues in the humanities were not enthusiastic. As Widmann has so aptly described the reaction, "the computer is either seen as an electronic wizard, about to yank us wawling and screaming into the brave new world of a post-technological society...or the computer is envisioned as a giant Coke machine...or the computer is perceived as a threat to the established order...." (1)

But experience has proven the computer neither wizard nor threat, and less like a Coke machine than a pinball game. Student fascination with making the bell ring has been put to good use in a series of English programs in BASIC which supplement rather than supplant standard teaching methods. This happy outcome, however, was reached through a somewhat painful process of experimentation which may be instructive to other teachers at small colleges.

Ironically, the very first set of programs I designed has been the most demonstrably successful. There are three reasons: the simplicity of the programs, the four years worth of debugging, and the evolution of a satisfactory method of integrating computer and class work. I used student assistants to develop these programs for a senior class popularly known as the "cram course" because its purpose is to prepare senior English majors to take a comprehensive examination in their field. The course reviews genre development through various periods and movements; students must, of course, learn to define key terms and major authors, develop a firm sense of chronology, and be able to show familiarity with a reasonable selection of significant works. Three computer programs help the student meet these goals.

CHRONO was originally designed for a history course. The student selects a file listing between 10 and 25 items in chronological order and decides how many items to attempt to arrange correctly. The computer scrambles the selected set and

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

prints a numbered list of items, then asks "Which came first?" and so on. This drill has several advantages: it shows students how well they understand chronology, it allows for progressive difficulty since it is much easier to do a program when the items are presented in small batches than when the whole file appears at once, and it stresses the relative order of events rather than exact dates. Thus a student of the novel must know that Tom Jones comes before Vanity Fair, but need not memorize either date of publication. This program has worked well from the beginning; we have rewritten files and directions, but students have remained happy with the original program.

ENGLISH is a short answer program. It asks questions - such as "What is the oldest anthology of English poetry called?" - which expect brief answers. Although the computer is programmed to accept up to three variant answers, the student must input one of those answers absolutely right. Initial student reaction was often frustration, since putting a "the" in where the computer didn't expect it or misspelling a word would make the answer wrong. So I added an optional crib sheet to the program. The student may call up a list of the answers before starting a file. It is still necessary to type in the complete brief answer, but the crib sheet ensures that the student does know the correct spelling of the answer as well as what it is. Because the answers are never more than a word or so long, students do not feel too frustrated by the typing. Most of them ask for the crib sheet the first time they run a given file, look up whatever items they feel uncertain about, and then run the program a second time to get a run with all the answers right.

MATCH is similar to ENGLISH, but the students are asked to identify relatively long chunks of material, such as quotations. The crib sheet numbers the answers and lists author, work, and sometimes the speaker or other identifying name. The student may input the number of the correct response or any one of the parts of it. Most students use the numbers, but some apparently feel that typing in the author's name helps to reinforce the information and produces a run suitable for use as notes, particularly if they are running the program a second time and trying to answer without the crib sheet. MATCH was initially plagued by technical difficulties and clerical errors in the typing of long quotations into the files, but it now runs satisfactorily and students like it for reviewing poetry.

ENGLISH originally printed humorous digs when students entered a wrong answer; student reaction was so positive that I eventually added an INSULT routine to all of my programs. These routines randomly select remarks, insert the student's name or nom de computer, and print them following some but not all mistakes.

The first year of the senior programs I was hoping for a spectacular improvement in student scoring on the departmental comprehensive. Students who take the cram course average about 10 (out of 100) points higher than students who do the examination on their own, and I had always supposed that this difference was due to the daily quizzes which forced them to review material. The computer programs replaced those quizzes; one run was due every class meeting. That first year kept me and my best student assistant run ragged getting the files on the system ahead of student use. (A total of 29 different files are currently available for the three programs.) Students did prefer running the computer programs to taking daily quizzes once they got used to the computer itself, and they said they were learning a lot from the programs, although running programs in addition to assigned readings made the work load pretty heavy. When the test results came in, however, I was disappointed: my students averaged the same 10 points higher as before. All that effort had about the same effect as constant testing in class.

But the use of the programs had advantages in other ways. Class time was freed for in-depth discussion instead of perpetual fact-stuffing. Students enjoyed doing the programs. At first I thought of them as tests and told the students they could run the programs as often as they wanted, and that I would substitute their computer run grades for the comprehensive test grade in determining final standing in the course. All the students did all the programs and a pattern developed: a small group would go down to the computer room and run the day's program jointly; they would then study together; finally each individual would run the program again to get a complete run to hand in. This cooperation seemed to make studying more agreeable, so I encouraged such practices. Many students spontaneously began writing notes on their completed computer runs.

Now I require the programs as "homework" and grade it on the following basis: a correct simple run is worth a C, a run with some sensible notes is worth a B, and a well annotated run is an A. Since the homework is 2/3 of the course grade, a student who works conscientiously will obtain an excellent average. I no longer assign specific secondary sources; the students buy a standard literature handbook and use earlier texts. Under the new system it is easier for students to study most where they are weakest. The computer programs are now so well accepted that some students who are not enrolled in the course have used them to prepare for the comprehensive.

Although the first year was nightmarish, at this point the existence of the computer programs saves me a great deal of class and quiz preparation time. I feel, however, that the biggest advantage is that the students come to think that "beating the computer" is fun, so they learn the factual material more willingly.

My initial success encouraged me to continue. As a linguistics teacher, I was fascinated by the possible application of transformational-generative grammar to computer programs; as a composition teacher, I was delighted at the thought of turning the computer to use drilling on such old bugaboos as spelling, punctuation, and bibliographical format. Two outstanding students who shared my interests worked on English programming as part of independent study projects. The results were excellent in terms of the intellectual development of the young programmers, but I was unable to use their work.

Robert Simon wrote two programs to my specifications. The one devoted to spelling utilized about 300 commonly misspelled words, sentences using the words, and comments on how to remember the spelling. The test format presented each word with asterisks in place of the most troublesome letters. Since all the student needed to type in was the missing letters, typing difficulties were minimized. The program never reinforced incorrect spelling by showing a word wrong, but it rang the bell and printed the entire word correctly when the student input was correct.

This program was fantastic. Unfortunately, one master file held all the information. The program would never run for very long at a time, and it annoyed students greatly to start over again just when they were getting the hang of things. Furthermore, the file was so complicated that the programmer himself had to be called in to make the most minor changes or corrections of clerical errors. After several months of frustrating tryouts I quietly gave up and wrote a much simpler program myself. It utilizes 16 separate files of two types, and while a person cannot run many files without running out of computer time, it is always possible to run full files. This program operates just as its prototype did, but students work through one segment at a time. Students using the NEWSPELL program this year are enthusiastic and I think I learned an important lesson: it is a mistake to let someone write a program which is too complicated to alter easily when splitting the program into smaller units can make it easier to write and use.

Bob Simon's second program was also amazing. It generated sentences randomly and effectively. Although it used a relatively limited number of syntactic patterns, the sentences produced by the program were nearly always not only correct but fairly sensible, because there was an elaborate mathematical system for matching semantic components. This program was supposed to help me teach transformational grammar. A student user could determine the probability of certain constructions occuring in a given run. I tried the program out for one semester. A couple of the students were intrigued that the computer could generate sentences. Most of the class was confused by the necessity for deciding that there should be, say, a 1 in 3 chance that the modal auxiliary would be present in the verb phrase. The program did not require enough active thought and input from the students, and although different sections were assigned at different times in the course, the program was not in use often enough to get students accustomed to the computer itself. Students will not enjoy CAI unless they are quite at ease with the mechanics of using a terminal.

While that program failed to provide enough for the student to do, another ambitious young programmer fell into the opposite trap. Cynthia Farina developed an elaborate program designed to teach footnote and bibliography format. All the students who tried it thought it took forever to run and asked just one question: why don't you program the computer to do it for me if I type in the information?

I have concluded that English students who become interested in computer programming should be encouraged to do linguistic or literary projects of their own, but that English teachers may get better results if they write their own programs.

I have myself written two new programs currently being tried out in my freshman composition sections. First there is the spelling program described briefly above. Comments on that have been favorable, although I noticed that many students did not bother to ask for information on how to spell words they didn't know, so I fixed the program to print the explanation automatically when a mistake is made. One evening school student who works with computers on the job has dramatically improved his spelling; he says he now understands when to check his dictionary. Several other weak spellers seem to be improving, although with not quite 40 students involved I obviously cannot offer statistical evidence. Still, the programs seem to be helpful and student response has been good enough to interest a couple of my colleagues in referring their problem spellers.

My second set of freshman programs was written last summer under a small grant from the Wagner Fund, part of the Ursinus endowment. They teach punctuation, using the same kind of sentence generating approach as the earlier linguistics project.

Five separate programs draw on a master library which contains subprograms generating such segments of the sentence as the verb auxiliary and the noun modifier. When the student runs a program, he or she is given a crib sheet with five answers explaining which mark of punctuation would be correct and the reason why. Then the program randomly generates and scrambles the order of sentences with structures matching the explanations. The student punctuates the sentences one at a time, with wrong answers corrected. No incorrectly punctuated sentence is ever shown; the program does print the sentence correctly after the student inputs the right answer number. The programs cover all the single marks of punctuation.

Since my previous experience with freshmen at the computer had not been very successful, at first I started using these new programs in a rather gingerly way. Our current departmental standards allot 20 points per composition to a mechanics grade. I got user numbers for all the students, held open sessions at the terminals to show them how to use the computer, and then told them that they could earn the full 20 points for mechanics on any composition by handing in a computer run with the essay. In addition, I promised to substitute an A+ for the student's lowest grade of the semester if a complete set of the 16 spelling files and the 5 punctuation programs was handed in with a paragraph of intelligent commentary.

Most of my freshmen tried the programs out. They were used most heavily by the best students, who often seem to find them intrinsically interesting as well as a good grade hedge, and by the worst students, who would lose 15 to 20 points per composition otherwise. Second semester, I required computer runs and that students mark sentences patterned after the computer runs in their compositions. This procedure effectively elicited greater variety in sentence structure as well as better punctuation.

The punctuation programs are needed by more students than are the spelling programs. They generate simple, compound, and complex sentences which are grammatical but not necessarily sensible. For example, here are some sentences from a run:

```
Swimming all over America, the analyst has loved some
    television stars cheerfully.
That describes Ellen:  solid and speckled.
My pair of slippers had been appearing purple;  moreover,
    the watch had remained clean.
The vampires wondered, "When must this heiress exercise
    today?"
```

Such sentences amuse most students, and they do force learners to punctuate according to the structure of the sentence rather than by meaning or feel. A few students, especially a native speaker of Spanish, have complained about particular sentences, often those using verbs of the "elect, choose, consider" type. I expect to revise that section shortly, and the programs do need minor revisions to eliminate occasional nongrammatical or ambiguous sentences. But the only serious technical difficulty was that the original programs allowed the student to do a set of 10 sentences, then ask for more, but the computer would often run out of time before the second set was finished. Now most of these programs run a fixed number of sentences and end, so the whole program must be run again to get more. The same sentences never appear twice, so the programs can be reused as frequently as a student desires.

Students who run the punctuation programs have seemed more willing to experiment with complicated sentences and advanced punctuation in their papers, and even in its first semester of use I saw a clear difference in student writing which I attribute to the computer: many writers who previously were unwilling to use any marks except periods and commas are developing ease and facility with colons, semi-colons, and dashes. I therefore intend to continue making these programs obligatory in my composition classes next year, and to require students to write standard sentences of the same grammatical pattern as computer-generated sentences. I believe I will be able to teach punctuation more effectively by gearing my class presentations specifically to separate computer assignments rather than having the runs trickle in over the whole course of the semester.

I also plan to modify the punctuation series slightly for use in my linguistics course, where the students will be asked to analyze the syntax of the computer-generated sentences as homework. Thus the program will be a source of raw material rather than a drill sergeant, although since many advanced students still need work on punctuation I intend to keep the punctuation questions in the program.

A major study of CAI has concluded that "The millions of dollars spent for many massive projects, in computer-based instruction, for example, have thus far left little tangible impact on what actually goes on in institutions of higher learning." (2) My experience at Ursinus can hardly be described as a "massive project," but the computer has continuing tangible impact on what actually goes on in my classes. The impact may even be spreading slowly into other English classes. Although most of my colleagues regard my enthusiasm for the computer as an eccentricity bordering upon betrayal of the humanities to the enemy, two of them have volunteered to run my programs and have referred students to them. We do not have enough terminal facilities to assign all freshmen computer work anyway, but I do expect that in the future numbers of remedial cases and also science-oriented freshmen from other classes will use the assignments on request. Students in my own classes do gain an understanding of the computer as servant to the mind which may, in fact, be a significant learning experience in itself.

I feel that humanities teachers may indeed utilize the computer effectively by using enough programs to accustom themselves and their students to the technique, maximizing the required student thought input and minimizing the required student typing input, integrating computer materials with class work, and retaining the reasoned perspective and seasonable humor typical of the humanities field at its best.

Notes and References

(1) R. L. Widmann. "Trends in Computer Applications to Literature." *Computers and the Humanities*, Vol. 9, pp. 231-232.

(2) John Fralick Rockart and Michael S. Scott Morton. *Computers and the Learning Process in Higher Education: A Report Prepared for the Carnegie Commission on Higher Education*. New York: McGraw Hill Book Co., 1975, p. 273.

AN UNDERGRADUATE COURSE IN COMPUTER-AIDED MUSICAL COMPOSITION UTILIZING TIME-SHARED BASIC

G. Clark Dehne
Department of Chemistry
(614) 236-6106

and

Bert A. Anderson
Conservatory of Music
(614) 236-6471

Capital University
Columbus, OH 43209

Abstract

For the past seven years, the authors have conducted an intensive four-week course on computers and music. This course has no prerequisites and is open to the musician and non-musician alike. It is a learn by doing approach and utilizes BASIC in a time-sharing environment. This paper is a report on the nature, success and problems of this experiment, and our hopes for the future.

Key Words: Computers and Music; BASIC and Music

Background

Since 1968, Capital University has been on a 4-1-4 (four months - one month - four months) calendar, the significant feature of which is the setting aside of the month of January as an Interim for a total immersion experience by students and faculty alike in non-traditional (or normally non-available) subjects. Although the total enrollment of Capital is under 3000 students, the University is composed of five separate colleges; a College of Arts and Sciences, a Conservatory of Music, a School of Nursing, a Law School, and a Graduate School of Administration. Only the latter two offer post-baccalaureate degrees and are not on the 4-1-4 calendar.

One of the challenges of the Interim period for a professor is creating a special course. The authors, one a chemist with an interest in music and computing, and the other, a musician with a theory and composition background, combined forces for the first time in Interim '71. Our proposal was to offer a course on computer techniques in music open to any interested students.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

Initial Limitations

We began our experiment despite some very clear problems and limitations:

1. The application of computers to music was well established (1-4). However, these applications were clearly too advanced for the beginning undergraduate student.
2. Neither of us had direct experience in this area. However, we felt that we could learn from one another as the course went along and, hopefully, there would be a true sharing of progress among students and faculty.
3. Our resources were severely limited. Capital had no in-house computer system. Educational computing capabilities had just been obtained by the funding of a hook-up to a commercial time-sharing service (G. E. Mark I) through an NSF grant to the science division under its COSIP program.

Given these limitations, and only four weeks, intensive as they may be, a question remained — what could the students really accomplish after having learned the essentials of programming and some modern compositional techniques along with any individual remedial work in mathematics and music theory? Accordingly, we were convinced that the music produced by the students would be uneven in "quality." Thus we decided that the purpose of the course should be to supply an intensive introductory experience, not necessarily produce "the best" music. We felt that if the student utilized the basic techniques offered in the course and was satisfied that what was produced was personally satisfying — or sounded as expected — then the experience was successful.

Language Choice

The choice of a programming language was easy. We felt the language had to be powerful, yet particularly for music students, easy enough so that it could be mastered (at least at an elementary level) in an intensive two-week period. The remaining two weeks could thus be set aside for the student to write, execute and score a programming porject of his own design. Further, text processing would be essential for ease of I/O operations in a musical environment. Finally, the time-sharing service offered only two languages of sufficient power —FORTRAN and BASIC. BASIC was the obvious choice for non-programmers.

There was another aspect dictated by hardware limitations. Because we were tied to a time-sharing situation, we could not implement any of the highly sophisticated compositional systems such as MUSIC IV. The nature of this limitation imposed a do-it-yourself programming approach for both students and faculty alike. We also, did not know which way the university would move in terms of computing capability in the future, which meant the programs should be portable.

Since the original offering of this course, Capital has moved to buying both educational and administrative computer time from the Battelle Memorial Institute CDC Cyber 73/6500 system. At the present time we have a remote batch terminal and nine interactive terminals (the latter a mix of teletypes, CRT's, DEC writers, and a Tektronix graphics unit) but still no "in-house" system. Fortunately, for our course, the switchover was not a significant problem. BASIC was available on the system and little recoding needed to be done. The students were unaffected by subsequent changes.

Nature of the Course

The original intent of the course was to address, at an undergraduate level, the two main trusts of modern music;

1. the impact of computers;
2. the electronic generation and manipulation of sound.

For pedagogical reasons, we felt that:

1. a hands-on, do-it-yourself approach was most desirable;
2. the course should be open to any student who had an interest in the field and was willing to work and learn.

This meant the course could have no prerequisites and would be an elective for curricular credit. Thus we would have to face the reality of a wide range of student backgrounds coupled with the imposed limitations of time and resources.

The First Two Weeks

The first two weeks are devoted to a mixture of lecture and practice on the fundamentals of programming and compositional techniques susceptible to computer enhancement and manipulation. For several years, we also included an introduction to electronic techniques, utilizing the Conservatory's electronic studio, but it became obvious that there was too much material to adequately cover in one month's time. Students in our Interim course can use the studio if they are knowledgeable and desire to do so, but that is no longer a requirement.

In addition to straightforward lectures on the elements of the BASIC language, considerable time is also devoted to algorithms for compositional techniques and their programming. These include such things as:

1. Pitch serialization and manipulation,
2. Rhythm serialization and manipulation,
3. General unput/output coding so that the student can communicate in simple musical terms (such as octave-letter notation for the well-tempered scale) rather than dealing in numbers.

During the period of time when the course included electronic realization of the student's work, this period was also devoted to a discussion of various scale structures, pitch/frequency relationships, and routines for moving back and forth between the two domains.

Also during this period the class is given a set of simple programming exercises. These include two programs that are already written. They need only to be typed in and executed. Thus, the students become a little less ill-at-ease in dealing with the machine, and gain some insights into both programming and coding techniques. The students are asked to have, at the end of this period, a small single voice composition that has been computer-developed. This composition is to be performed for the class.

During this "performance" the programming techniques are discussed for each piece, and the musical content is discussed. This serves as essential feed-back to the student, since some are quite disappointed with their results, and we can aid them in avoiding similar pitfalls during the second two weeks. Also, during these first two weeks, the students are introduced to some canned programs the authors have made available on the system. They are encouraged to use them for this first trial. At the conclusion of this period, we counsel with the students. We may recommend they team up with another individual in the class (trying to pair musical knowledge with programming expertise) and discuss the nature of their final project.

The Last Two Weeks

The last two weeks are totally devoted to student completion of a project (individual or team) which is then performed publicly on the last day of class. We have always allowed musical analysis to be a valid area for this project, coupled with a report rather than a performance. So far, no student has elected that option. During this period of time, the authors are always available for consultation and help. In some cases this can mean assisting in a rewrite of an entire program the student has been working on, after a week of floundering with infinite loops or coding a set of decisions that yields no valid answer. The authors have also used this time to write new programs for general use.

This period of time can be the most frustrating for the student. They learn very quickly that an executable program can generate a tremendous amount of data in a short period of time. This must then be scored and played before they find out what it really sounds like. If the student doesn't like the results, he usually can adjust the program accordingly and try again. Hopefully, this occurs near the end of the third week, so the student has time to clean up his programs and get something more desirable. Often this first try comes as late as a day or two before the final performance.

At this point time becomes a very significant factor, and the results may be considerably below the student's expectations. This problem magnifies the wide range of experiences already present among our students, and coupled with the fact that the score must often be sight-read by performers who are students themselves, results in a "concert" that is very uneven (the score may not even be written for the instruments that are available that day) nevertheless, the majority of our students have indicated that they did produce something valid. They only wished to have more time to perfect their work. This problem will be addressed in more detail later.

Evaluation

In evaluating our experience with this course, it seems reasonable to describe the work along three main lines:

1. The growth of the course in terms of content change in reaction to student expectations and capabilities,
2. By presenting examples of canned programs,
3. By presenting examples of student projects.

Although the three are interdependent, they do appear separable for purposes of this report.

Changes in Content

During the early offerings of this course, we concentrated almost totally on serialization of pitch and rhythm for several reasons:

1. It is a 20th century technique;
2. It is relatively easy to program;
3. It demands that the composer have some understanding of his material and an overall sense of direction.

Since that time, our concern has been heavily weighted towards pitch handling techniques. This is because:

1. A rather simple integer code can handle just about any vertical or horizontal manipulation.
2. A rhythmic code that is simple and easily serialized cannot be metered, except in a heavy-handed way, and vice-versa.
3. The students are quite happy if they can program something that primarily handles pitch manipulations, perhaps including some computer designation of rhythm.

We also concentrated on electronic realization of the composition because:

1. It represents a significant advance in music within the past twenty years;
2. There is a great deal of student interest in the field;

3. The instructors wanted to build this area within the Conservatory.

As noted before, the simple requirements of time, coupled with availability of equipment and numbers of students, made the inclusion of the latter educationally and logistically impractical. Further, we found that students concentrated on electronic realization and the computer aspects were not really being utilized. Thus, electronic music was made a separate course, still open to anyone, but spread over an entire semester, and our Interim course, since 1976, has concentrated on computer techniques only.

Two other significant changes have occurred since the early offerings. Because of the wide range of student abilities, we moved from a requirement of totally student developed software to making available a wider range of canned programs. We also realized that most students did not prefer the more easily programmed serial approach to composition on which we initially concentrated. However, programming other styles often requires much more time and expertise than could be legitimately expected of students in a first course. Accordingly, several complete programs were written, and another from the literature was added to aid those students who still had programming problems in achieving completion of a project in the last two weeks.

Sample Canned Progams

Programs used in the course include the following (Note 1):

1. FUGUE — a program that can generate two simple themes (or pitch rows) and then manipulate them, both horizontally and vertically using transpositions, inversions, etc., up to a total of four voices. The user selects the number of voices and their ranges — everything else is set. Rhythmic values generated by the program depend upon the intervals contained in the themes. The results are usually atonal and quite complex but occasionally satisfying. The basic raw material can be easily changed.

2. BABBITT — a simple serial program that constructs a Babbitt square of any row and prints it out. Students normally write a program to generate their own row and then feed the results to BABBITT in order to instantly expand their material.

3. CNTRPT — a progam which writes a counterpoint to a given thematic idea. Most of the rules of the 16th-century style are encoded and where there is a choice of paths a probabilistic switch is included so that the same original line can result in a different line of counterpoint. The user decides on which note of the given theme the counterpoint is to begin and its intervallic relationship.

4. HARMONY -- an adaptation of the MUSIC program available in their text(5). This program takes any melody and constructs one-on-one tonal SATB harmony using a pre-established preference table of triads and rules to ensure maximum movement of voices within established ranges. We have modified it to allow more freedom in the range of the bass voice, the input to be a melody in any key and added our own input/output routines.

All of the programs are fully documented for the students so that they are free to modify them as they see fit. In fact, modification is encouraged. Several students have done so, and the best of their work has been incorporated into later versions.

Student Projects

Regarding student projects, we find that they can be divided into three main classes:

1. Student programs that involve the computer generation of all material from some interactive starting point and may also include occasional interaction as the process continues;

2. Computer manipulation of previously developed musical material, such as folk songs, using available programs on the system;

3. Computer manipulation of non-musical data by student-written data processing programs.

Computer Generated Material

The first class often represents team efforts, where a science-oriented student works with a musician to develop a compositional idea into a single large program. However, there have been exceptional individual efforts. Some significant student accomplishments have included:

1. A mathematical construction of five balls on a billiard table, each ball representing a pre-established vertical combination of tones — then setting one ball in motion. The program solved, according to the laws of physics, the position of each ball as a function of time. This space-time relationship was then encoded by strict rules into a tone-time relationship for the chosen instrumentation.

2. The mathematical construction of a finite harp in a finite box. The program had the operator fire a "bullet" into the box with a selected vector and momentum. The bullet was allowed to bounce back and forth in the box, reporting when it struck a string and with what specific force. Momentum losses were taken into account, so that a complete composition could be built out of several "shots." This was executed using the precise pitch, time and dynamic data the program generated.

3. One group took a unique approach to the programming in musical form of the Fibinocci series. A starting pitch was given, as well as the total range, and the mathematics took over from there. Electronic techniques were used to smooth the discrete nature of the equation.

4. Several people have used a computer game approach in which the computer is programmed to play a game such as Nim or Blackjack. The various moves are encoded pitch-wise and occasionally time-wise, depending upon the rapidity of response of the two players.

5. A program has been written that develops a two-part modal canon of arbitrary length and constructs both lines according to appropriate rules from a simple two or more note "initializer." The idea is that the second voice must mimic the first, and the voices must follow certain harmonic rules with respect to each other. Thus, once an initial sequence is supplied, both melody and counterpoint can be developed by the program.

Computer manipulation of musical material

The second class of compositions is by far the most common. Here the students have tended to use available programs for their own purposes, either through modifications in the programs or through unique forms of realization. Some examples of this approach are:

1. A piece written in five movements by the FUGUE program — yet executed using a ten tone division of the octave rather than the usual twelve. Further, all other aspects of the composition were base ten rather than twelve as is standard Western practice.
2. A tonal set of themes generated by a student-written program which were then fed to CNTRPT and HARMONY. HARMONY was modified to include more movement in the voices than the standard block form the program generates. This was then scored for a bell choir and performed accordingly.
3. An organ piece built around student-composed serial themes, which were manipulated by BABBITT. BABBITT was moditied to include the computer selection of variants which were used for both horizontal and veritcal construction.

   In many of these cases, the students decided to take considerable liberty with the print-out so their choice of material reflected their original conception of the piece.

Computer Manipulation of non-musical material

The last grouping of compositions includes many varied approaches. Examples include:

1. The encoding of a textbook master chess game, with each square of the board assigned a specific pitch. Every square over which the piece moved became part of the row, and each pair of moves represented a measure. Special motifs were established for events such as capture, check and castling. The piece was performed as a recorder duet.
2. A team of two students encoded a year's weather data for the city assigning thematic material to such things as temperature readings and barometric pressures. It included rhythmic material for such items as rain, snow or thunder. This piece was never performed since it required a full 16-piece orchestra, and the scoring was never completed in time.
3. One of the more interesting electronic approaches was the computer selection of listings from the telephone directory. The dialing of those numbers and the recording of the ensuing conversations afforded the basis of composition. The material was electronically modified to add interest.

We have chosen for presentation projects that utilized the computer to the greater extent. In most cases, these applications have dealt with computer manipulation of material rather than generation. This is the nature of a course that is limited because of time and varied student backgrounds. However, we find that the majority of music produced has been acceptable and student surveys indicate that participants agree as to quality.

Summary

Overall, we have found the course successful. It is intensive and difficult, but the motivated student, regardless of background, succeeds, and that is a major goal. In terms of introducing twentieth-century techniques, we find some students like it and some do not. However, those music students with an interest in composition carry their work forward and utilize computer techniques in more advanced works. Finally, with the introduction of more tonal programs the satisfaction level of students has increased.

Looking back, the course has been demanding (but fun) for students and faculty alike. We have introduced the computer to over 100 students in the past seven years as a valid compositional tool. We have done it in such a way that they themselves can at least appreciate it. In most cases, students understand what is actually going on. No pretense has been made about the sophistication of the programs or student compositions; yet the students feel that they have done valid, acceptable work. Further, they have had a chance to hear their works performed in public. They are proud of what they have accomplished and justly so. Many of them remark that it was an experience they would go through again, although with a different point of view.

Future Plans

The nagging problem of time remains, and the authors have been given a sabbatical leave for Interim '78 to review past accomplishments and what we should do in the future. Our plans for the future include:

1. The possibility of applying pattern recognition techniques to musical analysis;
2. The possibility of applying graphics to I/O operations so that the output may be represented in score form;
3. The debugging and coalescing of the best of both our own and student programs into a coherent package usable by students in our course and others as well;
4. The determination of the best format for carrying on this kind of experience.

Other colleges have related computers and music at the undergraduate level (see e.g. 6 - 9, and other CCUC proceedings.) To our knowledge, however, our approach to computers and music has not been reported. There is no doubt in our mind that this approach is workable, and it has been successful on our campus. In addition to being used in promotional literature for recruitment, our continued efforts have become an expectation of both students and administration.

At the present writing, the only open questions are whether or not the course should be offered as a semester course, and what should be the guidelines for student enrollment. These decisions will determine the nature of the course content, although it is expected BASIC will still be the language for this offering. BASIC has proven itself to be a viable medium for undergraduate use in musical applications of computer processing for novices. We look forward to the future.

References

1. Foerster, Heinz von, and Beauchamp, James, eds., Music by Computers, John Wiley and Sons, N.Y. (1969).

2. Lefkoff, G., ed., Papers from the West Virginia University Conference on Computer Applications in Music, West Viriginia University Library, Morgantown (1967).

3. Lincoln, Harry B., ed., The Computer and Music, Cornell University Press, Ithaca (1970).

4. Matthews, Max V., etal, Technology of Computer Music, MIT Press, Cambridge (1969).

5. Kemeny, John G., and Kurtz, Thomas E., Basic Programming, 2nd edition, John Wiley & Sons, New York (1971).

6. Clough, John, "Computer Sound-Generation: An Educational Resource at Oberlin College," Proceedings, CCUC, University of Iowa, Iowa City (1970).

7. Killam, R. N., " Computer-Aided Analysis of Music Used in Undergraduate Music Theory," Proceedings, CCUC, COMPUTE, Hanover, (1971).

8. Hultberg, W. Earle and Hultberg, Mary Lou, "CAI in Music Theory: Paradigms: Potential: Problems," Proceedings, CCUC, Claremont Colleges, Claremont (1973).

9. Thieme, D. M., "The Applicability of Computer Assisted Instruction in College Music Theory," Am Mus Tchr, 25, N4, 18 (1976).

Note

1. Individuals desiring listings of these programs and further documentation are asked to contact Professor Dehne.

A NEW APPROACH TO
COMPUTER ASSISTED INSTRUCTION IN MUSIC THEORY

Richard L. Hamilton, M.M., and Dan W. Scott, Ph.D.

North Texas State University, Denton, Texas, USA

Introduction:

Computer Assisted Instruction (CAI) offers an effective method for presenting drill and practice in music theory. North Texas State University (NTSU) has developed a system which uses a unique group of components. These are a versatile and economic computer sound generation system and a simple yet effective curriculum management and presentation program. The two are independent but combine to form an effective and efficient method of teaching basic ear-training skills.

Every musician needs to acquire the ability to distinguish and identify musical sounds. The study of music is unusual among academic subjects in that the basic skills cannot be taught solely from a book. The particular combination of computer hardware and software described here was developed to provide students at NTSU with a drill and practice facility which would help them to develop their abilities in a structured manner.

North Texas State University has one of the largest music schools in the United States, with an enrollment of over 1700 music students. Nine hundred undergraduates enroll in some part of the two-year core music theory program each semester. All undergraduate music majors are required to pass a barrier examination in sight-singing and ear-training.

The ear-training skills required of music students are best developed through an instructional program designed to fit each student's needs. Such a program must include at least four things: (1) sound--the student needs to hear musical sound combinations; (2) immediate feedback--he needs to know whether he has identified a sound correctly before he forgets that sound; (3) patience--it normally takes many repetitions before a student can accurately and consistently identify sounds; and (4) individualization--each student's weak points need to receive extra attention.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

Individual personal instruction would undoubtedly be the ideal solution to this problem, but the economics of such an approach are prohibitive. The CAI program at NTSU provides all four of the above requirements economically.

Two separate but related efforts have been brought together to implement this music CAI system. The first was the design of a general score notation suitable for keyboard entry and its implementation on a microprocessor-based automatic music generation system. The Automatic Music System (AMUS) developed by Dan W. Scott is comprised of specialized hardware and microprocessor programs. In addition to its use in this CAI application, the AMUS can be used as an aid to composition, as a performing instrument or as a means for an amateur to explore and enjoy music. The second of these efforts was the development by Richard L. Hamilton of a special-purpose curriculum driver program for the host, or time-sharing, computer. This program is especially suited to drill and practice through the AMUS.[1]

Emphasis in this application is on drill and practice; the introduction of concepts is left to the instructor. This system provides a resource, musical drill and practice, which most students have not had access to--and does so without overburdening the instructor.

AMUS:

The Automatic Music System plays multiple-part harmonic music of varying amplitude with accurate timing and pitch.[2] Articulation, metronome, and other performance parameters are specified by the user. The user provides a text (a "score") written in ordinary typewriter characters by means of a simple terminal. This score is translated as it is read by a Motorola M6800 microprocessor computer system into pitch, amplitude, and timing control information for the electronic instrument, MUSOR, which is attached to the computer. The score may be typed in each time it is to be defined, it may be stored on a tape cassette, it may be stored or generated as text by a BASIC time-sharing (or other) remote computer facility (the "host computer"), or it may be text in a Computer-Aided-Instruction system which uses the host computer.

Development of AMUS involved (1) the specification of a new and extended scoring/performance notation; (2) the invention of a novel digital-to-analog waveform generator; and (3) the development of an elaborate computer program for score translation and for performance control.

AMUS differs from previous cheaper computer methods used for music generation in its quality of sound, including harmony and articulation, and in the complex software provided. It differs from previous more expensive implementations in the lesser quality of its sound with regard to timbre and amplitude control, and in the simpler score/performance notation, suited to the typewriter keyboard.

The MUSOR instrument contains 1 to 7 voices (5 for CAI use), with waveforms determined for each voice by a set of 16 fixed resistors. The articulation is under computer program control, as determined by the score. Timing is provided through computer program interrupts at the rate of 1760 per second. The MUSOR hardware is especially designed to be inexpensive, trading hardware expense for software complexity to reduce the material cost of the system. The expense has been increased, on the other hand, by a great emphasis on hardware reliability, the ideal being no maintenance.

Figure 1, on the next page, is a diagram of the hardware units which comprise the Automatic Music System, including the host computer.

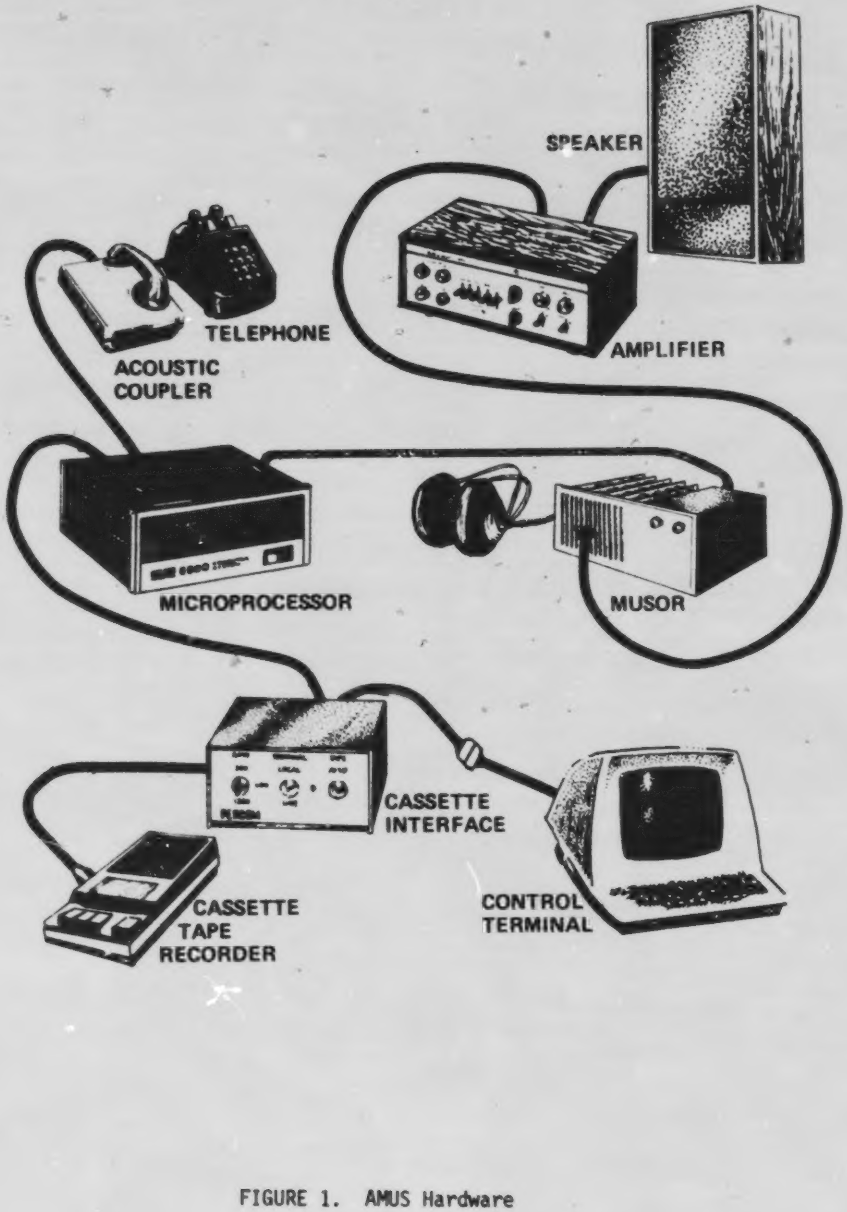

SPEAKER
TELEPHONE
ACOUSTIC COUPLER
AMPLIFIER
MICROPROCESSOR
MUSOR
CASSETTE INTERFACE
CASSETTE TAPE RECORDER
CONTROL TERMINAL

FIGURE 1. AMUS Hardware

AMUS receives scores and commands from either the local control terminal or from a remote time-sharing host computer. In order to handle a variety of score symbols as well as a wide range of text to be displayed to students, the AMUS program of the microprocessor has three modes of operation--text mode, score mode, and terminal mode. Certain commands change the mode of interpretation of input.

In the text mode, anything typed on the control terminal is sent to the remote computer and anything received from the remote computer is displayed on the control terminal. AMUS takes no other action until it receives a question mark (?) or a percent sign (%) from either source. Receipt of a question mark causes the score which was last read in and translated to be performed immediately; students may thereby cause a replay of the music. The percent sign signals AMUS that the next characters may be some kind of directive. If no valid text mode directive is found, the program takes no action and continues to read and display characters, remaining in the text mode.

If either of the directives %S or %SC is detected, AMUS deletes them from the display and moves into the score translating mode. In the score mode the input characters are translated into MUSOR control information for later performance. Score characters which are received from the remote host computer are normally not display to the student.

The score mode directives in general differ from the text mode directives. Table 1, below, gives the valid directives for the AMUS microprocessor in the text and score modes.

Table 1.

Directives--Text and Score Modes

%S	Begin score, go to score mode.
%SC	Continue previous score, go to score mode.
%P	Begin playing, stay in present mode.
%P*i*	Begin playing at measure *i*, stay in the same mode.
?	Begin playing, go to text mode.

Directives--Text Mode

%CS	Copy translated score to cassette tape.
%LS	Load translated score from cassette tape.
%X	Go to terminal mode.
%[	Copy arbitrary text into microprocessor memory.
%]	Transmit text from memory to remote system.

Directives--Score Mode

%E	End score and go to text mode.
%I*i*,*j*	Set attack time to *i*, steady state time to *j* units.
%M*i*	Set tempo to *i* units. (%M3904 = mm. Q=72.)
%K*x*	Set key signature to *x*.
%T*i*	Transpose following score by *i* semitones (*i* = signed integer).
i%O*j*	Set the voice *i* octave to *j* (middle c octave code is 4).
i%L*j*, *i*%N*j*, *i*%.*j*	Set articulation types. Voice = *i*; amplitude code of sustained portion of note = *j*.
%W	Start displaying score from host computer on the control terminal.
%W-1	Stop displaying score from host computer on the control terminal.
/	Measure marker used by %P.
/: :/	Beginning (/:) and end (:/) of section to be repeated (copied).
( )	Beginning ("(") and end (")") of comment to be ignored.

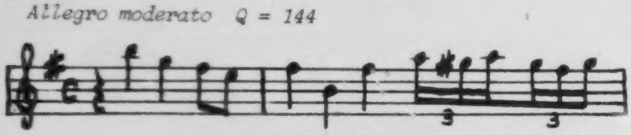
To leave the score mode a question mark can be typed, which starts playing the score and also gives control to the text mode. %E also transfers control to the text mode, but without playing the score.

The terminal mode is entered from the text mode by means of the %X directive. In the terminal mode, the system reacts exactly as a usual computer terminal. All text, including the text and score mode directives, is transparently sent from the control terminal to the phone line and vice versa without processing. The terminal mode is exited by pushing a reset key to return to the text mode. The terminal mode is convenient for looking at a stored score without actually translating it.

In addition to these modes, special convenience features are provided for logging on with a remote host computer.

An example of AMUS score notation is shown in Figure 2, below.

Score Example



```
%S  %K# (OR %KG)  %M2000 (Q=144)
RQ  BUQ  GQ  FE  EE  /  FQ  BDQ  FUQ  A3S  G#3S  A3S  G#3S  F3S  G#3S  /  (...
```

Figure 2. Excerpt score from the Sinfonia of Handel's Messiah.

The usual format for notes is: voice number, pitch/rest designator, accidentals for pitch modification from the key signature, octave shift specifier, triplet specifier, and articulation specifier. Each of these component specifiers is optional or has a default value. Most of the score specifiers are described briefly in the paragraphs below.

1 2 3 4 5 6 7 *Voice designator.* Must be the first character of a note. If the voice designator is missing, the symbol 1 is understood by default.

A B C D E F G R *Pitch designator.* As the first character of a note (possibly preceded by voice designator), it represents the pitch in a scale or, if "R", a rest.

+ P *Pitch modifier.* Synonyms for one semitone sharp.

X *Pitch modifier.* Two semitones sharp (double sharp).

B - *Pitch modifier.* One semitone flat.

Z *Pitch modifier.* Natural.

Accidentals (as above) normally follow a pitch designator and override the key signature. When more than one is used in a note the effect is cumulative, starting with natural. Note that accidentals do not apply through the measure; they effect a change only in the note that they are attached to. The pitch modifier symbols are also used in specifying key signatures with %K.

U D *Octave shift, up or down, respectively.* Shifts the current and all following notes for the voice up or down one octave. If more than one specifier is used the effect is cumulative.

W H Q E S T 4 *Duration specifier.* These characters stand for whole, half, quarter, eighth, sixteenth, thirty-second, and sixty-fourth durations. They may be combined for additive effect.

. *Dot duration designation.* A dot (period) following a duration in a note adds half of the duration to the left of the dot to the duration of the note.

3 *Triplet designation.* The triplet specifier must precede the duration specifiers to which it applies. Its occurrence in a specification makes the duration specifiers which follow it have 2/3 of their indicated values.

M *Duration subtraction.* The value of the note duration specification which follows M in the note is subtracted from the duration which has preceded the occurrence of M in the note. M resets the dot and triplet designations, so that specifiers which follow M are normally duple.

L N , *Articulation specifiers.* Legato, not-legato (normal), and emphasis articulation, respectively.

Curriculum Driver:

In July 1977 Professor Rosemary Killam began the implementation of a CAI program in music theory at NTSU. At that time two packages were available on the Hewlett-Packard 2000 time-sharing computer used at NTSU for CAI. These were: the Instructional Dialogue Facility (IDF) and PILOT.

IDF is aimed toward the user who has no computer expertise. It functions in an interactive manner both with the student and with the instructor who is developing a curriculum. It prompts the instructor for texts, questions, answers, and responses. It provides the student with texts, questions, and hints. Student answers are evaluated according to the patterns supplied by the instructor, and a limited form of branching is permitted. IDF also stores student responses and statistics.

Entering a lesson into IDF is simple for the instructor, but any variation from the set pattern can cause problems, and the random presentation of questions to students can only be accomplished with great difficulty. While IDF is useful for many applications (it was used for the music theory course at NTSU before the new curriculum driver was ready), it is not well-suited for the music theory curriculum, especially when the requirements described in this paper's introduction are considered.

The alternative available on the Hewlett-Packard 2000, PILOT, has a number of desirable features for the instructor: more flexible formats and more compact means to enter instructional material. The instructor types in simple commands which are interpreted and executed when the lesson is presented to the student. The commands are easily learned and easy to use. The main drawback of PILOT for the evaluation and development of curriculum is that the Hewlett-Packard version does not keep records of student performance data. Also, there is no concise way to implement repetitive drills.

The NTSU package developed by Richard L. Hamilton provides for data collection. It uses a format similar to PILOT, but it includes additional features necessary for drill and practice: concise entry, random presentation order, and three methods to judge student success. The package is similar to IDF in that it includes an enrollment procedure and complete data collection capabilities. All student responses are recorded for analysis, and a condensed file of statistical information is maintained for immediate reference.[1]

Students are given a suggested progression of courses and lessons but they can easily change this order to fit their own needs. A weekly report is generated which allows teachers to monitor their students' progress. Teachers can easily add lessons for special purposes.

Commands in the entry language used by the instructor are provided to control the presentation of the lesson and to evaluate immediately the student's progress. The driver can repeat material depending upon this evaluation. Repetitive structures are simple for the instructor to use. Randomization of the sequence of questions enables repetition to be concealed from the student; in addition, provision is made for random transpositions of scores ($R). The extent of repetition is controlled by evaluation of student success as the lesson progresses. A summary of the commands of the curriculum driver is given below. Each line begins with a one- to three-digit number and ends with a return.

T: The lesson is to type out the text which follows the colon (:).

A: Wait for an answer (reply) from the student.

M: Match the student's latest reply with the text following the colon. Remove blanks and question marks (this allows the student to request repetitions of the music performance without being penalized).

J: Jump in the lesson to the statement whose number follows the colon.

JN: Jump only if the student's last response was incorrect.

JY: Jump only if the student's last response was correct.

TN: Type the text following the colon if the last response was incorrect.

TY: Type the text following the colon if the last response was correct.

P: Pause for the number of seconds given after the colon.

E: End of the lesson; causes termination of the lesson. This must be the last statement in the lesson.

Q: *n<p\a\c\t>* This is the format for questions. *n* specifies the number of questions which follow in the form *<question\correct answer>*. The instructor uses as many lines as are necessary for his convenience. The parameters of the Q command are summarized as follows:

n gives the number of questions in this group of questions.

p specifies the presentation method. R causes the questions of the group to be presented in random order, while D steps through the examples in the order given by the instructor.

a specifies the method of evaluation of the student's answer. D means that the student must match the answer provided by the instructor exactly; I means that no evaluation will be done.

c specifies the criterion for termination of presentation of the group of questions. A*i* means to continue presenting questions from the group until the student gets *i* questions right on his first answer to each question. P*j/k* means to present questions until the student gets *j* right on the first trial. If he has not gotten *j* right before seeing *k* questions from the group, then the counts are restarted for the presentation of more questions from the group, until the criteron is satisfied. After passing the criterion the student continues with whatever follows the group of questions. A third option, N, steps the student through the group of questions once.

t specifies the number of trials, 1 to 9, that a student gets to answer a question when it is presented each time. Note that this feature has no effect on the *c* = A or P criteria, which count only first-trial success.

RY: Right-answer reply for all questions of the group. A default reply exists, provided by the curriculum driver, which can be restored by using RY: with no text.

RN: Wrong-answer reply for all questions of the group. A default reply exists, provided by the curriculum driver, which can be restored by using RN: with no text.

F: Failure reply for the question group. The text which follows the colon specifies the reply for an incorrect answer on the last trial allowed for the question. A default reply exists, provided by the curriculum driver, which can be restored by using F: with no text.

H: Specifies the text for a hint. Any number of consecutive H: statements are saved and will be presented, as a group, each time the student types //HINT. If no text follows the colon, then all previous hints of the group are eliminated.

In addition, there are abbreviations which manipulate strings of characters. A dollar sign ($) in any output string (whether question, response, or other text) signals one of several operations, depending upon the character which follows the dollar sign. $N is replaced in the string by the student's first name, which is obtained at the first of the lesson when it is presented. $A is replaced by the correct answer to the most recent question. $R generates $S $T*i*, where the generator automatically replaces *i* with a random number from -7 to 7 in value. This causes the score which follows to reflect a transposition by a random number of semitones; the value of *i* is recorded with the student's response.

The criterion feature of the Q command allows the teacher to control quantitatively the conditions of presentation. This feature is particularly useful when the teacher needs to present a limited number of examples, perhaps 3 or 4, but wants to insure that the student can hear each example as many times as he needs for comprehension. For instance, if the criterion provided by the instructor specifies ten correct answers, the student cycles through the statements of the Q group until he achieves the necessary ten correct answers.

The random presentation order and random transposition options allow the patterns of question repetition to be concealed. Thus the teacher can specify a large amount of curriculum with a minimum of text entry. A text editor is provided with the curriculum driver which allows duplication of lines and permits easy editing within lines. These features expedite curriculum entry.

The Q command and the general economy of the system have greatly reduced file storage space as well as entry time. Lessons originally created in IDF which have subsequently been converted to the new system are now stored in approximately one-fifth the file space of the original and were entered in much less time. The sample lesson below shows the basic format and illustrates the use of the Q command and the format for questions and answers.

```
10  T:  HERE'S A LESSON TO HELP YOU LEARN INTERVALS IN MAJOR TRIADS.
20  T:  YOU'LL HEAR A TRIAD, THEN AN INTERVAL.
30  T:
40  T: YOU TYPE THE INTERVAL'S NOTES BY TRIAD MEMBER.
50  T:
60  T:     TYPE U (FOR UP) BETWEEN NUMBERS IF THE INTERVALS ASCENDS;
70  T:     TYPE D (FOR DOWN) BETWEEN NUMBERS IS THE INTERVAL DESCENDS.
80  T:
90  H: TRIAD MEMBERS ARE 1 OR 3 OR 5.
100 Q:  3<R\D\P4/5\3>
110  <$R 4CD 3GD 2C EH,  1R 2R 3R 4RQ  2CH,  EH,  ? \ 1U3 >
```

(continued...)

(Lesson example, continued from the preceding page.)

```
120 <sR  4CD 3GD 2C EH,  1R 2R 3R 4RQ  2EH,  CH ? \ 3D1>
130 <sR  4CD 3C  2E GH,  1R 2R 3R 4RQ  2CH,  GH ? \ 1U5>
9999  E;
```

Evaluation:

After a semester of operation with a single AMUS terminal, the results are encouraging. Subjectively, the student response to the use of CAI is favorable. Objectively, the first generation of the system was used in an experiment with two sections of the twelve freshman music theory classes at NTSU in the fall of 1977. These two sections had the lowest placement scores on a standard examination. Both were taught by the same instructor. Students in one class used the computer drill-and-practice facility while the others did not. On the mid-term examination the students who used the facility had a median score of 20 percentage points higher in ear-training skills than those who did not; at the end of the semester, a significantly larger proportion passed, and with higher grades, than the other group.[3]

Prospects:

Future plans include: display of a limited amount of music notation on control terminals, addition of question generation capabilities to the driver, an increased number of AMUS terminals available for CAI and experimentation, and modifications of MUSOR to widen the range of timbral possibilities and articulation. Computer identification of the pitches of notes hummed by the student is being considered. Longer-range ideas include the development of score entry via piano keyboard and the eventual production of a stand-alone version of the entire system using microprocessors to eliminate the time-sharing host computer. Such a system might offer four AMUS terminals, each processing the student interactions, with lessons and statistics kept on a fifth processor with disk memory.

The situation at NTSU also provides an opportunity to study a large population of musicians and to observe how they can best learn the special skills they must obtain to become proficient in their art.

REFERENCES

1. Most of the statistical routines are the work of Jack Chao of the North Texas State University Computer Sciences Department.

2. Scott, Dan W. "AMUS, The Automatic Music System". Presented at the meeting of the Fort Worth Chapter of the Association for Computing Machinery, September 13, 1977.

3. Professor Rosemary Killam of NTSU is preparing a more detailed study of the results of using CAI drill-and-practice in the music theory program at NTSU.

A STRATEGY FOR USING COMPUTER PROGRAMS IN TEACHING CARTOGRAPHY

James W. Cerny

Department of Geography
University of New Hampshire
Durham, New Hampshire 03824
(603) 862-1719

Introduction.

With the quantitative and qualitative improvements that have come in computing facilities during the past decade, there has been a consequent transformation of the content of some geography courses that deal with map making. The selective use of computer exercises, particularly at the undergraduate level, can improve the balance between necessary practice and repetitive work. Baumann (1975) and Malmstrom (1973), for example, discuss the use of particular program systems that they have developed. The emphasis here is somewhat different, with discussion focused on a portable, flexible collection of individual mapping programs, used more to teach map design than to teach data collection and analysis.

At the University of New Hampshire the introductory cartography course deals with the design and drafting of small-scale and medium-scale special purpose maps, of the type commonly found in theses and scholarly publications, what cartographers call thematic maps. The course is project oriented: It starts with a discussion of some unconventional transformations of space (mental maps, cartograms) and progresses to a consideration of conventional map projections, dot mapping, proportional point-symbol mapping, choropleth mapping, isarithmic mapping, reprography, and other topics. The students are expected both to understand the principles and techniques of map

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

design involved and to learn the rudiments of drafting. The students bring a variety of backgrounds to the course, with half of them typically non-geography majors. However, very few students have used a computer before taking the course.

The use of the computer is twofold: (1) for projects where time saving is important, to eliminate repetition or involved calculation, and (2) as an adjunct or alternative, sometimes for drill and sometimes for comparison with manual methods. Students are taught to view the computer as an additional piece of equipment that can be used in map making.

Strategy.

In seeking computer programs for use in a thematic cartography course or some other mapping course, there are several existing sources that might be considered because they are either free or low cost. Tobler (1970) describes and lists a number of FORTRAN programs, MacDougall (1976) discusses some techniques and lists some FORTRAN segments that could form the basis of other programs, and Wittick (1976) discusses a large FORTRAN package that incorporates many analyses that have in the past been distributed separately through the Geography Program Exchange (GPE). However, none of these sources place a strong emphasis on portability. Many of the programs are written for large CDC and IBM computers and it may be difficult to implement them on other systems. There are also, almost inevitably it seems, problems of missing routines and documentation and, perhaps most important, many programs are not intended for use by beginning students.

The strategy described here, for developing a library of programs for class use, was necessary to make the best of the resources available. The collection was assembled gradually, but the general requirements were fixed at the start. First, the programs had to be interactive (terminal-oriented), such that novice users could understand the dialog. It was possible to capitalize on some programs that were already available, but the majority were written especially for use in the course. Most of them were originally implemented on an IBM 360/50 under CALL/OS, but now all of them are running on a DECsystem-10. Since they are interactive, each student can correct minor errors on the spot, get quick results, and nothing needs to be batched by the instructor. To minimize problems in following directions, a sample or model run of each program, with annotations, is usually posted for students to look at. Second, most of the programs are in BASIC because they are easy to write and easy to convert if necessary. Although there is only an interim BASIC language standard in existence at present (Committee X3J2, 1976), much of the portability problem with BASIC can be solved if a limited subset of BASIC is used (Dunnagen, 1976; Lientz, 1976). For example, several of these programs have been converted to run on a PDP-11, with only a small number of straight-forward changes needed. Most differences involve reading and writing disk files, so disk file access has been generally avoided, even at some sacrifice in program elegance. An added advantage of BASIC is the ease with which the instructor can take responsibility for maintaining many of the programs himself and modifying them to suit his own needs, or even adding to the collection. While it might seem attractive to be freed from these responsibilities, in fact it often proves difficult to get someone else to do the programming the way you want it, when you want it.

The Program Collection.

Table I is a list of the programs involved. In addition, we have a number of large batch-oriented programs, such as SYMAP, which are not really suited to an introductory course. Not every program is used every semester and greater reliance is placed on some than others. A typical sequence of usage might be as follows. REALGY is used as a demonstration to get students to learn the mechanics of calling up a program from one of the public libraries. It asks the students some amusing, but trivial, location questions. SHAPE1 is used to allow the students to verify a hand computation of the Boyce-Clark shape measure. MPROJ1 is used as a drill after they have made a globe and several map projections, to see whether they understand the basic properties of the Sinusoidal and Mercator. FLANRY is available, for optional use, to compute scaled circle radii for proportional point-symbol maps. Much of the focus is on the programs for choropleth mapping, since the computer does that well and it is possible to illustrate several topics (shading, class interval selection) with them. This group has been described earlier (Cerny, 1975), but involves CMAP to make choropleth maps, CMAPDC for documentation, NHMAP and NHCART for base map files, and to a much lesser extent SCANCV and SCANMK for file construction. Using these programs and files students are able to make choropleth maps of New Hampshire (10 counties), without having to prepare the base maps. CLASS1 is used to compute several kinds of data class limits (equal, arithmetic, geometric, reciprocal) and to do some of the calculations preliminary to CMAP. SHIFT is also used as a data preparation program, to do shift-share calculations that relate change or growth of constituent areas to the expected change for the larger area. Finally, TSADC and TSAPLT are used to produce some simple trend surface maps.

The program size given in the table is the approximate length in characters, including comments. The origin of the programs is either UNH, Beloit, or Dartmouth. All of those with the source indicated as UNH are written by the author and are available to educational users in their DEC-10 version as either a listing, or on magnetic tape or DECtape. Write for details to Applications Librarian, Computer Services, University of New Hampshire, Durham, New Hampshire, 03824.

Observations and Additions.

The student reaction to the programs is uniformly favorable. The novelty of using the computer doubtless accounts for some interest, but the students seem to find the assignments interesting and worthwhile. Although some students have more trouble interacting with the computer than others do, they are all able to carry out the assignments. None of the assignments normally take more than 30 or 40 minutes at a terminal.

The thought of monitoring student usage and individual program usage is a recurring one. On our DEC-10 it is not practical to get individual usage reports on either students or programs. On our IBM system we did get monthly reports on individual students, but it was of limited usefulness. Typically a few students would use no computer time in a given month and a few students as much as 30 or 40 dollars worth. Some students would either not use the computer at all or would do so on a friend's account number. Other students would play games, in addition to their cartographic work. A course budget estimate could be prepared on the basis of such monitoring, but little else.

The library is under regular refinement and expansion. The greatest need is for a fast, interactive contouring program that will accept irregularly gridded data. In some years SYMAP has been used in a batch set-up, but it has not been very satisfactory, since the students end up mechanically preparing the data and are somewhat removed from the result. If a Tektronix-type graphics terminal or a digitizer was used in the course, some additional programs would be needed to do such things as generate three-dimensional surface plots and rectify and scale coordinates. Depending on the course and time available, there are many other computer programs available now in batch form that it would be challenging to set-up in interactive form; for example, programs could be added for drawing population pyramids, cartograms, Thiessen polygons, and map projections.

Literature Cited.

Baumann, P. R., (1975), "A Computer-Based Instructional System on Maps," _Journal on Geography_, Vol. 74, pp. 159-166.

Cerny, J. W., (1976), "CMAPS: A Basic Language Program for Choropleth Mapping," _Creative Computing_, Vol. 2, No. 6, pp. 72-74.

Committee X3J2, (1976), _Proposed American National Standard for Minimal BASIC_. New York: American National Standards Institute.

Dunnagan, T. (ed.), (1976), "Technical Transfer Guidelines," *CONDUIT Guides: Aids for Transfer of Instructional Computing*. Iowa City(IA): CONDUIT.

Lientz, B. P., (1976), "A Comparative Evaluation of Versions of BASIC," *Communications of the ACM*, Vol. 19, pp. 175-181.

MacDougall, E. B., (1976), *Computer Programming for Spatial Problems*. London: Edward Arnold.

Malmstrom, V. H., (1973), "The Computer as a Tool of Map Analysis," *Proceedings*, Fourth Conference on Computers in the Undergraduate Curricula, June 18-29, Claremont, CA.

Tobler, W. R. (ed.), (1970), *Selected Computer Programs*. Ann Arbor: Department of Geography, University of Michigan

Wittick, R., (1976), "A Geographic Computer System for the Analysis and Graphic Depiction of Spatial Point Data," *Geographical Analysis*, Vol. 8, pp. 319-328.

TABLE I. Programs in the Cartography Collection.

Program Name	Source	Language	Size	Function
CARTLB	UNH	BASIC	5625	Directory of programs.
CLASS1	UNH	BASIC	4985	Data class limits.
CMAP	UNH	BASIC	2985	Choropleth mapping.
CMAPDC	UNH	BASIC	4435	Documentation for CMAP.
FLANRY	UNH	BASIC	2165	Scale map circle radii.
MPROJ1	Beloit	FORTRAN	-na-	Sinusoidal & Mercator drill.
MPROJ2	Dart.	BASIC	16120	Projection properties drill.
MPROJ3	Dart.	BASIC	32750	Projection selection drill.
NHCART	UNH	Data	195	NH cartogram scan-file.
NHMAP	UNH	Data	180	NH normal base scan-file.
REALGY	UNH	BASIC	1225	Demonstration quiz.
SCANCV	UNH	BASIC	1540	Renumber scan-file areas.
SCANMK	UNH	BASIC	1995	Create scan-files.
SHAPE1	UNH	BASIC	1515	Boyce-Clark shape index.
SHAPE2	UNH	BASIC	1885	Center of gravity.
SHIFT	UNH	BASIC	5190	Shift-share calculations.
TSADC	UNH	BASIC	3610	Documentation for TSAPLT.
TSAPLT	UNH	BASIC	3975	Plot trend surfaces.

Source: Author.

A COMPUTER-INTEGRATED COURSE IN RESEARCH METHODS

Arthur St. George

The University of New Mexico

The development of the computer represents one of the major technological innovations of the last twenty years. Although colleges and universities have had access to these machines since the very beginning of their development, only up until the last few years only selected students, mainly those in the physical sciences and engineering, came into direct contact with the computer. The object of this contact was primarily to give the student an ability to utilize the services of a high speed calculating machine. Recently, however, attempts have been made to broaden student use of the computer so that business, social sciences, and humanities students are also exposed to the computer. More importantly, attempts have been made to enlarge the role played by the computer within the educational process, to make it a viable learning tool.

These developments owe their origin to several different technical and educational advances including an increase in faculty knowledge and appreciation of the computer, the development of simple computer languages and software packages, and the development and dissemination of interactive or time-sharing systems. These last two factors represent major forces leading to change. The development of simple computer languages has reduced significantly the time which both students and faculty need to spend in order to acquire an ability to communicate with the computer. The development and dissemination of time-sharing and remote terminal facilities has reduced wait-time for answers to almost zero; in fact, it is now common for the user to experience two-way communication or conversational computing. Two promising developments which will lead to even greater use of the computer are networking and microcomputers. The former allows a number of users, often geographically far apart, to have simultaneous access to a large central computing facility. When it becomes more economically feasible for smaller colleges and universities to invest in heavy minicomputers, departments will have their own computer for use whenever they choose as well as access to a larger, central university machine.

Although technological and educational advances in recent years have demonstrated the usefulness of the computer in the learning process, its use in a number of disciplines remains minimal with what appears to be slow but increasing popularity.[1] Sociology is one of these disciplines. Recent reviews of social science computing (Anderson, 1974; Goldsmid, 1976) list an impressive array of

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© 1978 CCUC

the various uses social scientists have made of computers, but relatively few of them involve its use as a pedagogical, as opposed to merely computational, tool. From published bibliographies such as these, and an examination of numerous primary documents listed therein, it appears that the pedagogical use of computers by sociologists is primarily restricted to courses such as introductory sociology.

Even here, however, the use of the computer is not totally integrated into class material but commonly is presented as a tool merely to reduce computation time. While this is perhaps a more legitimate perspective for introductory sociology, such is not the case for statistics or research methods courses. Again, published reports indicate that for these latter types of courses, the computer functions mainly to reduce the drudgery of statistical computation although in pursuit of this goal, students often learn a computer language. Erickson and Jacobsen (1973), for instance, present proposed changes in the teaching of statistics through the use of a computer. Specifically, they suggest teaching students sufficient FORTRAN to allow them to translate statistical formulas into FORTRAN programs. While this may indeed, as they argue, free students' time to be spent on more important issues, their emphasis remains with the efficiency and time saved by use of the computer. Thus, they state that "No longer does a research assistant need to spend hours at the calculator doing mechanical operations;" and "Programming knowledge greatly increases the efficiency of a research project in terms of man-hours and indirect costs." (Erickson and Jacobsen, 1973:99).

Merely teaching students how to program may solve time and efficiency problems, but it does little to contribute to their understanding of what computers are, how they work, their scope and limits, and most importantly, their diversity vis-à-vis uses by sociologists. In addition, this approach does not take full advantage of the computer's power to allow the instructor flexibility in creating, for example, a software package tailored to his course.

There are, however, some encouraging recent developments which demonstrate the flexibility and broad use of computers in the sociology curriculum. For instance, Wieting (1975) describes an application of computer simulation to teaching research methods. Using evaluation data derived from an experimental setting, he concludes that computer technology is superior to man-model simulation "since the technology provided greater facility in the very characteristics that were judged by students in this early period to be the most satisfactory about the form of the simulations." (1975:52) Van Valey and Brouilette (1974) suggest a "piggyback" course as a means of teaching students how to use the computer to solve statistical problems while also teaching them the computational basis of the solutions. They accomplish this by asking the students to take two courses simultaneously - the regular four-hour statistics course and a special, one-hour computer course. The courses are carefully integrated to teach both computational and data processing skills with the latter involving use of "real world" data. Elsewhere, Van Valey (1977) offers more detailed and concrete suggestions for courses using computers in an analytical mode. Finally, Brown et al. (1977:329) describe a "systematic approach for teaching undergraduate sociology students social research through student use of packaged computer programs while they carry out their own research projects." Their unique approach involves the use of the computer in the development of a questionnaire as well as subsequent data analysis.

Although the above uses of the computer are innovative and promising, it is clear that the computer is not as integrated into the course as it might be. This is especially true in terms of using the computer to alleviate the instructor of certain tasks such as examination administration and record keeping.

---

1. For a review of recent advances, the reader should consult the Proceedings of the First through the Eighth Annual Conference on Computers in the Undergraduate Curricula.

The present paper describes the efforts of the author to develop and implement an almost totally computer-integrated methods course and the pedagogical advantages of such a course. The rational for developing such a course evolved, partially, out of a concern to provide students with marketable and useful methodological skills consistent with contemporary sociological research practice. Prior to a discussion of other rationale, I shall give a relatively detailed description of the course.

Course Description

Sociology 481 is a relatively small (20 students per term) senior-level research methods course with two prerequisites: a course in elementary statistics and one in sociological applications of statistics. The course is intended to introduce students to research methodology and data analysis with an emphasis on applied techniques and problems. Most of the students have weak backgrounds in mathematics, the unusual student being the one with a single course in college algebra. The students' experience with computers is equally limited. The primary work of the course is a data-analytic paper in which the student is expected to use path analysis to analyze a small (no more than five variables) model. The student is expected to gain access via SPSS to a "canned" data-set residing on either tape or disk and using the regression program in SPSS plus an interactive path analysis program, perform the analysis.

A traditional research methods course usually emphasizes one of two basic approaches. First, the course may focus on the abstract scientific process of theory construction, the relation of theory to method, hypothesis formulation, operationalization, and so forth. The other approach features excessively detailed examinations of the various data collection techniques, burdened by the associated laborious field and laboratory work. Sociology 481 takes neither approach; rather, it introduces the student immediately to significant sociological data such as Stouffer's communism study, Dahl's radical right research, and other datasets selected from those available through the ICPSR (Inter-University Consortium for Political and Social Research) of the University of Michigan. This direct involvement with real datasets enhances the students' understanding of the theory, practices and use of survey statistics. It is true that this focus emphasizes data analysis to a greater extent than it does data creation. And, as I have heard suggested, it may be the case that this orientation of a methods course is more prevalent now than in the past. With the greater stress today on sociology being a science and its practitioners helping to accomplish this by quantifying everything and then "crunching" the numbers in a computer, this is probably true. Nonetheless, it is possible to strike a compromise such as the course described herein. Students in 481 are exposed to data generation methods and are required to gain experience with two of them, participant observation and interviewing. Unfortunately, time does not permit them to experience other methods such as experiments. Finally, although the course emphasizes data analysis rather than data generation, this does not mean an exclusion of theory. The principal data-analytic strategy, path analysis, was chosen because it forces the student to offer a theoretical rationale for the selection and ordering of variables.

To accomplish the primary work of the course, the semester is divided into three modules: 1) computer science; 2) "standard" methods; and 3) statistical analysis. Each module is approximately five weeks in length. The remainder of the course description will focus on these modules.

It is a central objective of the course to introduce students to the computer. This is done through both traditional classroom methods and experimental techniques. From the first meeting of the class, the students begin their experience by discovering that the syllabus, ordinarily distributed by the instructor, is stored in the computer and the first assignment is to retrieve it. Simple, task-oriented instructions are given on how to locate the computer center (most students have never been there) and operate the remote terminals. The syllabus informs the students that among other things, they are expected to go to the terminal at the beginning of each week and retrieve the assignment for that week. The computer is programmed to erase the old assignment and generate the new one at a pre-specified day and time.[2] These weekly assignments are additive and contribute, with a few exceptions, sequentially to the writing of the final paper. In addition, each student's grade on the assignment is entered by the student to determine the status of his weekly assignment grades.

The first module introduces the student to the basic idea of computer science, a programming language (BASIC), and a major software package (SPSS). The intention here is to give the students a conception of what computers are, how they work, what they can and cannot do, and how they can be of use to sociologists. The outline in Figure I describes the basic information necessary to bridge the gap between the computer scientist and the student.

Figure I

Outline of Course Material for
Computer Science Module

I. Hardware

  A. Central Processing Unit, Buffers, Arithmetic and Logic
  B. Peripheral Equipment
    1. Unit Record Equipment
      a. Keypunch and Verifier
      b. Card Machines - Reproducer and Interpreter
    2. Input Devices
      a. Card Reader, Tape, Disk, Mark-Sense, Paper Tape
      b. CRT
      c. Keypunch
      d. Other Remote Terminals
    3. Outpunch Devices
      a. CRT
      b. Other Remote Terminals
      c. Card Punch, Tape, Disk, Paper Tape, Printer
    4. Conversion Equipment
    5. External (Secondary or Auxiliary) Storage

II. Software

  A. Computer Languages
    1. Low Level Languages
      a. Assembly
      b. Machine
    2. High Level Languages
      a. BASIC
  B. Using Computer Languages
    1. Programming
      a. Flow Chart
      b. Coding
      c. Debugging
    2. Program
      a. Assembler
      b. Compiler
  C. Packages
    1. SPSS
  D. Input
  E. Output

---

[2] This and other sample programs are available from the author.

Since even the more basic concepts of computer technology and processing seem inordinately abstract to the average sociology student, special classroom techniques are used to convey the material. Central among these is a man-simulation model of how a computer functions. The students are assigned roles corresponding to units of a computer such as the supervisor language, memory, logic and arithmetic, I/0 buffers, and related peripheral devices. Then, with the guidance of the instructor, the students work through a very simple problem such as computing a sum and comparing it to a predetermined value. The students learn first-hand, through often amusing incidents, of a computer's need to have tight logic and precise timing.

The first several assignments in this module are directed toward increasing the students' familiarity and ease with both the card reader and terminal. This is accomplished by having the students retrieve, through batch mode, documentation on the computer center, available services and software, and details on the use of BASIC and SPSS relevant to their computer. In addition, they are given a tour of the computer center. For terminal experience, students are asked to engage the computer in a variety of shared-library games such as hangman, star-trek, and roulette. Following this, they begin to learn about computer languages and acquire specific skills in BASIC. Currently, their assignment at this point is to write a general-purpose program which will balance a checkbook. Since BASIC is a totally time-shared language, this enhances their experience with the terminal as well as producing a useful result. Finally, they begin work on the various ICPSR codebooks in an effort to select a moderately large (20-30) number of variables that represent a substantive area of interest they would like to explore, eventually for the final paper. The focus of work in the class at this point is on SPSS. It is taught initially as simply another language with specific advantages for social scientists. The students are thus able to transfer concepts, such as syntactical structure, from BASIC to SPSS. Their goal in the latter assignments of this module is the production of an analysis deck, consisting of a subset of variables and a subsample of respondents from one of the larger ICPSR studies. At this point, the students have a choice of running their data to their remote terminal library.

Thus, at the conclusion of the first module, students have a working knowledge of computer hardware and software, experience with the keypunch, card reader, and remote terminal, elementary programming in BASIC, and, through SPSS, the ability to retrieve and manipulate large datasets residing on tape or disk.

The second module covers more standard methodological material with a concentration on issues that are most relevant to the use of canned datasets, e.g., the relation of theory to research, hypothesis formulation, operationalization, index and scale construction. Even here, however, students must use the computer every week. At the very least, students must use the computer to access the week's assignment. However, some assignments are "conversational," i.e., not only does the student receive the assignment at the terminal, but he is quizzed on parts of the assignment and must respond at the terminal with the correct answer. These assignments structurally resemble the older, non-computer, programmed learning texts. For instance, one assignment in index construction presents the student with the concept a particular index is supposed to measure along with numerous possible items for inclusion in the index. After going through a conversational interaction on the degree of face validity of the various items, the student is given several crosstabs representing pairs of items he has selected. He is then quizzed on the interpretation of the tables, vis-à-vis a sufficient level of covariation to justify the items being included in the index. Inasmuch as this entire exercise takes place at the terminal, the student gains familiarity with the computer concomitantly with learning methodological concepts and techniques. Also, since this and other assignments all must be completed on the students' analysis deck, they contribute directly to the final paper.

The final module begins with a review of simple linear regression and goes through nonrecursive path analysis. The intermediate topics include a relatively

168

standard treatment of multiple regression, residual plot analysis, trend analysis, and dummy variables. The emphasis throughout is on interpretation of computer output rather than on the algebra of the techniques. In this module, the computer is used heavily for demonstration purposes as well as for student assignments. Since it is as convenient to produce twenty-five copies as it is to produce one copy of a demonstration run of, say, a five-variable multiple regression, most lectures are supplemented by the handout of an actual computer run using the technique being discussed. As might be expected, during this module students spend most of their awake hours either at the card reader or the terminal. Weekly assignments are a mixture of conversational and batch mode tasks. Thus, one assignment may simply present the student with a problem, such as interpreting a two-variable regression analysis from his analysis deck, and a week in which to solve it, while another assignment may involve a more complex "hands-on" task. In the latter case, one assignment asks them at the terminal to compare a stepwise with a hierarchical regression output. Using a shared-library dataset, a program asks specific questions about the results and, depending upon his answers, probes the student for better and more accurate answers. For path analysis, the student uses a modified version of the conversational program presented by Nygreen (1971). This excellent program allows students to enter linkages, compute paths, and revise the model, all while sitting at the terminal.

Pedagogical Issues

There are numerous models of teaching and most instructors have experimented with several of them at one time or another. As most instructors of methods courses can testify, the traditional lecture-based course is inadequate because it does not allow the students sufficient experience in doing methods. Historically, the response to this problem has been to require students to participate in all phases of a research project, usually survey in nature, from problem formulation and question writing through coding and tabulation of results. The recent and rapid advance of research instrumentation, specifically the computer, has eliminated the need to require students to spend valuable time on less valuable efforts such as coding. Along with instrumentation development has arisen the need to create and implement new models of teaching that most effectively utilize the rapid student feedback and efficient teacher monitoring made possible by modern technology. One of the more promising models, and one used in the course described here, is the discovery-learning model.

The discovery-learning model is based on the evident excitement and inspiration of those moments in the classroom or laboratory when teacher and student achieve together some significant insight into the nature of things. As described by Nuthall and Snook (1973), the discovery-learning model consists of loosely grouped set of ideas from a variety of psychological schools. They note, however, that the model "represents a new mode of relating the verv old humanist-Gestalt tradition in psychology to the practice of teaching." (Nuthall and Snook, 1973:60) Although there are various descriptions of the discovery-learning model, they all have in common the emphasis on the teacher not telling the students the principle or generalization or rule which they are supposed to learn. The primary pedagogical benefit of this model is summarized by Nuthall and Snook (1973:62): "When discovery has occurred, then knowledge gained in other ways, it is more easily transferable to other situations, and has a more important place in the student's mind."

In order for the "discovery" to be most effective and meaningful in the context of a methods course, it is necessary to highly structure the material presented while making it appear to the student that it is not highly structured. This is easily accomplished by what I describe as a computer-integrated course. In such a course, the computer is utilized to benefit both the student and the instructor and allows for an orderly progression of materials from "known" to "discovered". This process, central to the discovery-learning model, is paraphrased by Hendrix (1961) and Davis (1966):

> In the first stage pupils are asked to solve problems or discuss examples for which their prior knowledge is appropriate and adequate. In the second stage the problems or examples are changed, slowly or abruptly, so that the known procedures become either cumbersome or impossible to apply. During this stage the successful student "discovers" a new method or principle which enables him to handle the new problems more effeciently or correctly.

The computer-based weekly assignments function as these two stages.

The heavy use of the computer throughout the course, combined with the discovery-learning model, has clear pedagogical advantages over the more standard, traditional lecture-based course in which the computer is used, if at all, as an adjunct apparatus. Primary among these is the deliberate integration of sociological theory, methods, statistics, and computer science in a problem-solving framework. In performing a task in one of these areas, a student is never far away from the others and thus learns to view the research enterprise as a totality and not simply as unconnected tasks each to be approached in a cookbook fashion. Thus, for instance, in an assignment requiring the development of an index, the student must concern himself with conceptualization and operationalization, problems of internal consistency and validity checks which include the notion of covariation with perhaps a statistical test, the latter steps being done at the terminal using SPSS.

From the perspective of the instructor, there are numerous pedagogical and pragmatic advantages to the computer-integrated methods course. The author has noted the following specific advantages in his experience with both traditional and computer-integrated courses. First, use of the computer reduces the instructor's time spent on more mechanical activities. All too frequently the instructor becomes overly involved in activities not directly associated with teaching, i.e, he has the burden of assigning, grading, and giving students feedback on homework and examinations; helping students with their assignments; and conducting tutorial/remedial sessions. To a great extent the computer can perform these tasks, on an individual basis, as well or better than the instructor, for the computer can be programmed to be the world's most patient tutor and has the capability of interacting with the student as often and for as long as the student requires. Also, in the author's experience, this does not diminish the instructor's contact with the student but rather, makes it possible for the instructor-student interaction to be richer in the activities which instructors perform best: giving insights into difficult concepts, transmitting an understanding of abstract ideas, inspiring students and achieving behavioral objectives in an affective domain. Secondly, a computer-integrated course provides the instructor with the means to efficiently monitor the progress of students. Since the computer retains a master file of the students' progress throughout the course, it is possible at a glance to determine where and to what extent a particular student is experiencing difficulty. Thirdly, extensive use of the computer in the fashion outlined here results in a tight structure to the course. With the necessity of creating programs and assignments which are highly integrated and logically connected, the instructor thereby develops a sequence of tasks which follows the research process itself, thus giving the student an understanding and appreciation of the context in which individual tasks are performed. Fourthly, and of special relevance to methods courses, the computer facilitates the development and use of demonstration material. Once the nature of these demonstrations has been established, at any time in the future, they may be produced in a matter of seconds. Especially where statistical demonstrations are concerned, the computer aids in executing oftentimes dull and exacting tasks in a quick and efficient manner. In addition, these demonstrations have more meaning for the student if they are performed with actual datasets such as those described earlier. Finally, as a result of the computer science module early in the course, students learn a sufficient amount of knowledge and skills that their dependence on the instructor for assistance with computer problems is all but eliminated.

In teaching a computer-integrated methods course for five years, the author has noted the following major pedagogical advantages, which accrue to students: 1) it is possible to eliminate certain trivial statistical material which will probably never be used outside the class, e.g., computing correlations using a scatter diagram; 2) less attention to the mechanical and algebraic details of statistical calculations is required; 3) the computer can effectively provide concrete demonstrations of abstract statistical and methodological concepts which would otherwise be very difficult for the student to comprehend; 4) by freeing both student's and instructor's time from laborious, manual tasks, the computer acts to enhance student-instructor interaction; 5) use of the computer creates unusually high levels of student enthusiasm; 6) students with poor mathematical backgrounds are not prevented from understanding the intuitive basis of statistical concepts; 7) it is possible to concentrate on individual learning processes through both computer-assisted assignments and analysis of individually chosen datasets and variables; 8) because of the relatively fast feedback on problems, students are brought into closer contact with their data during the analysis process and it provides them with a powerful learning situation in which quick exploration of ideas can be done along alternate paths chosen by the student in an ongoing process of interaction with the computer.

Pragmatic Issues

The course described herein is unique in the sense that it attempts to integrate, to the highest degree possible, the computer with all aspects of a research methods course. Because this approach goes beyond most uses of the computer in such courses and because social scientists are becoming increasingly sophisticated consumers and users of computer hardware and software, it is useful here to describe some of the practical considerations that went into designing and implementing a computer-integrated methods course. While there are a number of general considerations in using the computer in the undergraduate curricula, most of these are discussed by Van Valey (1977) and thus will not be reiterated here. The following concerns will be of interest to those thinking of creating a computer-integrated course similar to the one described here.

As most computer users know or have experienced, hardware incompatibility is primarily responsible for the problems associated with transporting a program from one installation to another. The reason is simple: most programmers write the most efficient, optimal program possible and this usually means using techniques that are hardware-dependent and unique to their computer. To some extent, this is true of the programs described herein. The programs were written for and run on an IBM 360/67 with MVT and using HASP and ØS release 21.8. The time-sharing system is CALL-ØS. Including the datasets and programs, offline storage requirements are approximately 20 cylinders of a 2314 disk. The SPSS tasks which require processing of an entire dataset, such as one of the NORC General Surveys, require approximately 218K bytes of core, although an OSIRIS run requires only one-half of this amount.

The students have access to two types of terminals: (1) the computer center's Decwriters made by the Digital Equipment Corporation; or (2) the sociology department's Hewlitt-Packard 2640, a cathode ray tube type terminal. Both terminals operate at a rate of 30 characters per second (CPS). The center has approximately 60 such terminals and it is unusual for the students to have to wait more than 30 minutes for a free terminal. Although a TAB function reduces the problem, the main complaint about the Decwriters is the difficulty of spacing, which is especially important when using SPSS.

Turning from hardware to software considerations, a few words about the level of computer support services is in order. The university computer center's primary support for projects of this kind is limited to consultation on the choice of hardware (in this case our terminal) and assistance with the debugging of programs. In-depth advice on programming techniques or design is unusual and for the most part the user is expected to purchase the appropriate manuals and embark on a do-it-yourself adventure. Assistance with major software decisions, e.g., program module design or chaining of programs, would greatly facilitate projects of this nature.

Other sources of support such as financial or institutional would also enhance the attractiveness of undertaking the development of a computer-integrated course. In the author's case, neither source of support was more than token. Financial support from the university was limited to resources such as the purchase of the terminal, tapes, cards, disks and the like. When I began the planning of this project five years ago, institutional support for such efforts was meager; little social science computing was being done on campus and administrators beyond the departmental level were unfamiliar with and suspicious of the potential benefits. In other words, while the author received token praise for such work, he did not receive the most useful support: release time from teaching in which to work on the computer programs and course design. Without such support, only the more motivated or masochistic individuals will attempt such ambitious projects. Furthermore, from a cost efficiency point of view, the development of such a course without institutional support becomes nearly prohibitive. However, once the course is developed and the programs debugged, the cost efficiency factor is encouraging. The course allows the author to spend, up to a factor of three or more, additional time for students, research, or the like. In addition, from the students' perspective the course is structured such that they may move swiftly through the material thus maximizing the efficiency of their time spent on out-of-class work.

Finally, there are a number of more specific and concrete comments and suggestions that should be made. First, with regard to the length of time it took to develop the various computer programs and the expertise required, it is important to note that, in addition to the expertise of the person developing the course, these factors are somewhat dependent upon the conditions mentioned above, i.e., institutional support, and so forth. The author's background includes extensive education in physics, a discipline in which students early in their career learn the necessity of programming. In addition, the author spent several years employed as a software programmer for a university computer center. Although this is not the background of the average sociologist teaching methods courses, this level of expertise is not required to develop a course such as that described here. It is, of course, necessary to have a high degree of familiarity with computers, a programming language, and SPSS. For example, knowledge equivalent to that obtained from reading a book such as Donald Veldman's Fortran Programming for the Behavioral Sciences is sufficient. In addition, it is important to be very knowledgable of your local computer and specifically, I/O operations, the use of tapes and disks, and the time-sharing system. The basic rule should be to assume that you will receive little assistance from the computer center. Then, if you do receive such help it will be a bonus.

With institutional support in the form of release time, a person with the above-mentioned skills should be able to develop a computer-integrated course in two semesters. Because of the conditions described above, the course described here took approximately five semesters to develop. An additional factor which may influence development time is money. The University of New Mexico Computer Center provides nearly unlimited support for instructional and research (unfunded) uses of the computer. In universities or departments where this is not the case and "hard" money supports the computer facility, development time may be considerably slower.

To facilitate the development and dissemination of computer materials such as those described here, transportability is a crucial factor. This means that the developer must give consideration to the problem of balancing the ease of program design using local computer-dependent features with the potential problems of conversion faced by someone using the material on another computer. The majority of the programs described here are transportable, i.e., with very little re-writing they will function on most other computers. For the most part, they are written in BASIC and FORTRAN. The programs that handle more complex tasks such as intricate I/0 and auto-tutorial assignments are written in PL/I and Assembler language and are more machine dependent. Of course, other installations with roughly the same configuration as that described above can expect relatively fewer conversion problems. It should be pointed out that all of the programs used in the course could have been written in BASIC (and could, with some effort, be re-written in BASIC), but the author chose to integrate other languages to maximize effeciency and turnaround time. Finally, while this course uses SPSS as the primary analytical book, the concepts and techniques would work equally well with other packages such as OSIRIS, OMNITAB, or SAS.

Evaluation

Unfortunately, I do not have data of an experimental nature with which others could evaluate the usefulness or success of a computer-integrated methods course. However, other objective and subjective data do shed some light on these matters. First, there are the student evaluations of the course. At the beginning of the semester, most of the students were inexperienced with and afraid of the computer, especially as it related to working with numbers and statistics. By the end of the course, they had become competent data processors and many of them discovered that "doing" statistics could be fun. The following are illustrative of their response to the course:

> The course provided me with a variety of skills and knowledge which greatly increased my understanding and appreciation of the realities of social science research.
>
> I found this course one of the most interesting that I've taken at the university. It was also the most time consuming and mentally challenging.
>
> For the first time I learned something about how computers work. Now, things that used to be hard to do and confusing in working with the computer make sense.
>
> Notwithstanding the hard work and the difficult nature of the subject matter, Sociology 481, unequivocally is an excellent and beneficial course that provides and develops valuable information and procedures in marketable skills: organization, analysis, and interpretation of data and development of confidence in the utilization of computers.
>
> Although it cuts down on your ability to work with other class members, being able to work at your own pace with computer produced and scored assignments is a great help.

Additional evidence of favorable student response to the course comes from what I call the "advice to incoming students." At the conclusion of the semester I ask the class to write short statements of practical advice they would give to their friends taking the course the following semester. Then, at the beginning of the following semester I collate and hand out these comments to the incoming students. A few illustrative remarks from this "advice" follows:

> Along with the inevitable frustrations come benefits - take pride in knowing that your new skills rank you among a small minority of people who have any degree of facility with the computer or statistics.
>
> Last of all, make sure you enjoy this class. You'll work harder and have more fun than most of the classes in your college career.
>
> Don't count on the computer. The computer has a history of "crashing" the day before an assignment is due. Therefore, accept this as a fact of life and plan time accord-

> ingly. This suggestion can also include written assignments. Start all the assignments as soon as they are available at the terminal.
>
> He (the professor) is a strange man but he can be very helpful when there is no one else to turn to for comfort and consolation.

Additional evidence that the course is successful in providing students with marketable skills comes from the fact that many students report that as a result of taking the course they have secured employment in some research-related job. A number of students were pleasantly surprised to discover that employers were willing to pay them for their knowledge of survey analysis and SPSS. A final indicator of the course's increasing popularity is the fact that in the five years it has been offered, the enrollment of students from majors other than sociology has increased 400 per cent. In fact, several departments are now requiring the course for their graduate students. Apart from the students' response to the course, additional evidence of its acceptability and usefulness is the fact that faculty interest in the course has also increased dramatically since its inception. Not only have some faculty audited the course, but most have admitted that the course has been primarily responsible for a renewed interest in computing in the department.

Objective data also indicates that the students' use of the computer, as well as related instructional uses (e.g., development) has increased dramatically since the implementation of the computer-integrated course. In 1972, the year before the course was introduced, the sociology department's computer budget was $9,051. In 1973, this figure increased to $12,502. And, in 1976, the department spent over $95,000 for computer time! Adjusting these figures for increases in CPU charge and realizing that 75% of this money each year is spent on instructional uses, there is good reason to believe that the computer-integrated course has had an impact on student and faculty alike.

REFERENCES

ANDERSON, R. E. (1974) "Bibliography on Social Science Computing." Computing Reviews (July): 247-261.

BROWN, W., I. COOK, and C. UNKOVIC. (1977) "Effectively Teaching Undergraduate Social Research Via Computer Data Analysis." pp. 323-329 in Proceedings of the 1977 Conference on Computers in the Undergraduate Curricula. East Lansing, Michigan: Michigan State University, 1977.

DAVIS, R. B. (1966) "Discovery in the Teaching of Mathematics." pp. 114-128 in L. D. Schulman and E. R. Keisler (eds.) Learning by Discovery: A Critical Appraisal. Chicago: Rand McNally.

ERIKSON, M. L. and R. B. JACOBSEN (1973) "On Computer Applications and Statistics in Sociology: Toward the Passing Away of an Antiquated Technology." Teaching Sociology 1(October): 84-102.

GOLDSMID, C. (1976) "Key Resources in Teacher Development." Teaching Sociology (April): 339-378.

HENDRIX, G. (1961) "Learning by Discovery." The Mathematics Teacher 54: 290-299.

NUTHALL, G. and I. SNOOK (1973) "Contemporary Models of Teaching," pp. 47-76 in R. Travers (ed.) Second Handbook of Research on Teaching. Chicago: Rand McNally.

Proceedings of the Conference on Computers in the Undergraduate Curricula, Pullman, Washington: Washington State University, 1974.

Proceedings of the First Annual Conference on Computers in the Undergraduate Curricula (1970). Iowa City, Iowa.

Proceedings of Fourth Conference on Computers in the Undergraduate Curricula (1973). Claremont, California.

Proceedings of the 1975 Conference on Computers in the Undergraduate Curricula. Fort Worth, Texas: Texas Christian University, 1975.

Proceedings of the 1976 Conference on Computers in the Undergraduate Curricula. Binghamton, N.Y.: State University of New York at Binghamton, 1976.

Proceedings of the 1977 Conference on Computers in the Undergraduate Curricula. East Lansing, Michigan: Michagan State University, 1977.

Proceedings of the Second Annual Conference on Computers in the Undergraduate Curricula. Hanover, N.H. (1971).

Proceedings of Third Annual Conference on Computers in the Undergraduate Curricula (1972)

ROTH, J. (1966) "Hired Hand Research." American Sociologist, 1(August): 190-196.

WEITING, S. (1975) "Simulated Research Experiences for Use in Teaching Research Methods." Teaching Sociology, 3(October): 33-59.

VAN VALEY, T. and J. BROUILETTE "The 'Piggyback' Course as a Device for Teaching Statistics in Sociology." pp. 123-125 in Proceedings of the Conference on Computers in the Undergraduate Curricula, Pullman, Washington: Washington State University, 1974.

VAN VALEY, T. (1977) "The Computer and Doing Sociology." Teaching Sociology, 3(April): 277-293.

COMPUTER-ASSISTED HOMEWORK AND TEACHING EFFECTIVENESS: A CASE STUDY FROM INTRODUCTORY SOCIOLOGY[1]

George H. Conklin

Sweet Briar College

INTRODUCTION

The discipline of sociology today rests heavily on the results of a generation of research by scholars using empirical research. Old theories have been tested, confirmed or rejected as computers have made available to an increasing number of researchers and students reliable data-bases which previously were largely unavailable or too difficult for practical analysis. Further, it is not unknown that theories have to be modified when computer analysis reveals that data from studies tabulated before computers have proven in error.

Without at least some feel for the ways in which empirical data are analyzed, students are entirely cut off from reading social science articles from journals and from understanding even at the intuitive leval a large number of books in sociology as well. Students get into bad habits fast. The freshman who is introduced to sociology minus all social facts (or data) will be the senior who studies for comprehensive exams by reading an ever-increasing pile of low-level secondary sources. As Davis (1978:1) has put it, without social facts introductory courses "...consist of highfalutin but vague concepts, wars between academic armies mostly consisting of one soldier each, glittering generalities, and liberal or left wing ideological indoctrination. Most students come out of most sociology courses with little or no empirical information about the topic covered."

Davis concludes that students must be presented social facts at the same time they are given theory. He argues "I did not say we need facts only and I didn't say all courses should be primarily fact oriented. I do think it is scandalous that virtually all courses are virtually fact free (Davis 1978:2)." For Davis the natural unit of empirical sociological knowledge is the "dynamic three-to-five variable system." This is in fact correct; the problem remains

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

how to introduce a student at the beginning level to multi-variate analysis without turning the course into a methods course. This paper describes one effort to introduce students to linear multiple regression without ever mentioning such a horrible word. The paper will conclude with an evaluation of the effectiveness of the teaching program described in the appendix.

THE EXPERIMENT

The context used to explain multiple regression to the introductory student was the study of deviance, a chapter or topic in virtually all introductory sociology textbooks. Students who would be totally uninterested in linear multiple regression as an abstract concept find little difficulty understanding how multiple-factors may well be associated with variations in the crime rate. I usually approach the subject by saying simply that before a crime is committed or a juvenile becomes a juvenile delinquent, a long chain of events has probably taken place, each of which helps move a person closer to committing a crime. For example, living in a tough neighborhood may be one factor, but then to add poverty will only aggravate the situation. So if you add in enough factors, I conclude, nearly everyone would be forced to respond, and some might even respond by getting violent.

In keeping with the rules evolved earlier by the author on how to design programs for beginners (see Conklin 1976; 1977), the first part of the text of the program contains a paragraph orienting the student to the intellectual issues involved. Students are then introduced to several measures of poverty and how they correlate with crime (see the appendix). They are given the opportunity to see how changes in levels of poverty are associated with different rates of aggravated assault. At the end of the program, each student must consider all variables together in a regression equation first managed by the machine and finally as a "pencil and paper" exercise to be handed in. The reader is asked now to read the entire appendix to best understand the organization of the material as a student sees it. It should be noted that students overestimate the murder rates by a factor of 50 to 1, and the majority believe the past was safer than the present.

EVALUATION OF THE PROGRAM

The program described here is not designed to replace the usual textbooks, nor was it designed to replace lectures or discussions. Rather, the approach tries to add to the existing course exposure to the empirical side of sociology by dealing with specific social facts as part of homework. Computer-assisted homework (hereafter CAH) adds meat to existing courses without detracting from the usual lectures. To study the effects of the teaching program we designed a questionnaire to be administered the first day of class and the last day of class. The questions relating to the program described here are listed in Table 1. The answers circled are those considered "best" so a student who fills out the questions as circled would get a score of 8. Table 2 gives the analysis of responses "before" and "after," including the results of a control group consisting of another section of introductory sociology not using the programmed homework assignments.[2] Table 2 considers the differences in the classes as group, not the amount learned by an individual.

The selection of questions was based on two assumptions. First, we felt that students who did the CAH would have more factual knowledge than those who did not. Specifically, we felt that the fear of crime is such in American society that students would dramatically overestimate their chance of victimization, a fact borne out. Question 3, for example, gave an absurdly high range of values because pretests showed that it was the range which students felt "real." We would be happy, therefore, if those exposed to data lowered their answers compared to those who were in the control group. This was in fact what happened, as can be seen also in question 8. Secondly, however, we wanted to see if students who did the program could solve a simple algebraic equation.

Table 1

Pre-Test and Post-Test Questions (See text for explanation of answers)

1. Below is a list of several measures of success in tennis matches for three individual players. Through research we find that each measure requires a different weight. Please calculate the following answer

	Predictor X	Predictor Y	(fill in) Answer
Player 1	8	5	
Player 2	10	3	98
Player 3	3	2	FOR PLAYER 2

Use the formula: $8X + 6Y =$ answer.

2. It was found that 300 people in a population of 1,530 were sick in a given year. How many sick people per 1,000 does that work out to?

apx. 200 (fill in)

3. How many people per 1,000 per year do you think are victims of aggravated assault in the United States (1973 data)?

a. 300
[✓ b.] 50
c. 900
d. 5000

4. Who, in your opinion, is most likely to be a victim of a crime in the United States?

a. a high income individual with a lot to steal from
[✓ b.] a low income individual living in a moderate house.

5. A record club places an ad in a newspaper with a reply coupon which asks you if you have a telephone. How important do you feel this information is to the company placing the ad?

[✓ a.] very important
b. somewhat useful information which they might use
c. it is unimportant except that they want to call me up to sell me something.

6. Within American cities, the association between population density and the crime rate is

a. very strong
[✓ b.] moderate
[✓ c.] very weak.

7. How greatly do you fear being the victim of a crime in the coming year?

a. greatly fear
b. think about it quite a bit
[✓ c.] it might happen to me
[✓ d.] I don't worry much about crime right now.

8. According to F.B.I. data, what chance does an average citizen of the United States have of being a victim of aggravated assault in any one year at the present time in the Virginia-North Carolina area?

a. 1 chance in 1 million
[✓ b.] 3 chances per thousand
c. 55 chances per thousand
d. 1 person in a 100 will be a victim.

73.7 percent could do so before the class began, but nearly all (94.7 percent) could do so after. Five of the eight questions were statistically significant in the direction predicted. The control group showed no significant changes.

Table 3 is the result of a regression analysis of an achievement index which was formed by measuring how much a student learned. For example, if a student answered question 1 incorrectly on the pretest but got it right on the posttest, she received a score of 1. If she got it wrong after first getting it right, she got a score of -1. If she was consistent (right or wrong), a score of 0 would be assigned. For questions 3, 5, 6, 7 and 8, a student was assigned a score based on change. For example, if a student answered "d" for question 8 in the pretest, and a "c" for the posttest, this would result in a gain of 1. Individuals were summed over all 8 questions (there were no missing answers) and assigned a score for the index.

The regression equation reveals that exposure to the program was the largest factor in explaining change, while type of high school, etc. was not significant. However, the SAT in math was a correlate which entered second in the regression equation. The negative correlate shows that those who learned the most were those with the lowest board scores in math. This finding is important since it suggests that those who learned the most from CAH were those least well-prepared for college-level work. Item 3 reveals that high school performance helps predict college performance.

Since the achievement index in Table 3 measured only individual improvement, it is entirely possible that students with poor board scores simply had more to learn than those with better scores. The active and control groups were therefore examined separately by assigning a student a score based on the "right" or "wrong" answers as shown in Table I. A separate regression analysis was made on each of the four groups-- before and after for the active group; before and after for the controls.

The additional analysis revealed a strong positive correlation between the SAT math scores and a good score on the pretest. For the active group the correlate was a (statistically) significant .55; for the controls it was .44. But after doing the computer-assisted homework, the correlate between the SAT in math disappeared for the active group, dropping to only .09. With the control group, however, the correlate remained .40. In other words, the computer-assisted homework succeeded in bringing up the less able members of the class to the same achievement levels as those with higher ability.

DISCUSSION AND CONCLUSIONS

Students in sociology badly need exposure to social facts even at the introductory level. A useful way to accomplish this is not to discard the usual lecture, rather it is to supplement the "wars between academic armies" with carefully prepared computer-assisted homework assignments. CAH seems to have provided an opportunity for the less-able student to catch up to the better-prepared student to the point of actually evening out performance levels. Considering that the "cost" for this is only a half-hour on the computer (for this particular exercise) with no additional classroom time needed, the case for CAH is compelling. It will become even more important in future years as the loss of selectivity at colleges becomes even more severe and college professors will be required to teach those who now might not be admitted for higher education. Good students learn from CAH, but poor students learn enough to catch up.

There are no miracles in education. However, computer-assisted instruction does seem more efficient than lecturing in teaching concepts (see Lorber and Skinner 1977:87). Using the computer-assisted homework program described here, the author was able to add empirical material to an introductory level course without otherwise modifying the usual course.

Table 2

Before and After Scores on Achievement Evaluation

Item	Group	Before	After	Sig.*
1. Question 1: Proportion with Correct Answer	Active (N=19)	73.7	94.7	.1
	Control (N=17)	76.5	76.5	NS
2. Question 2: Proportion with Correct Answer	Active	36.1	52.6	NS
	Control	35.3	41.2	NS
3. Question 3: Mean of Scale	Active	2.2	1.4	.01
	Control	1.9	2.1	NS
4. Question 4: Proportion with Correct Answer	Active	89.5	89.5	NS
	Control	82.3	64.7	NS
5. Question 5: Proportion Responding A or B	Active	57.9	89.5	.05
	Control	70.5	70.5	NS
6. Question 6: Mean of Scale	Active	1.2	2.2	.001
	Control	1.4	1.5	NS
7. Question 7: Mean of Scale	Active	3.1	3.1	NS
	Control	2.8	3.1	NS
8. Question 8: Mean of Scale	Active	3.3	2.5	.05
	Control	3.4	3.4	NS

*Tests of significance are one way analysis of variance. Question 2 would be significant if other tests were used.

- -

Table 3

Regression Analysis of Achievement Index and Control Variables

Item	Zero Order	Mtple r
1. Experimental Factor (Control Group = 0; Active Group = 1)	.58	.58**
2. SAT in Math (Below 300=1; 300-399=2; 400-499=3; 500-599=4; 600 and on up=5)	-.16	.63*
3. High School Class Position (top quarter=1, etc.)	-.20	.69*
Items Not Included by Regression Program		
4. High School Type (public or private)		
5. Math Background (in years)		
6. SAT Score in English (coded as for item 2)		

**statistically significant at the .001 level
*addition statistically significant at the .1 level.

Appendix, Footnotes and References

```
          CRIME AND SOCIAL INDICATORS
          ----- --- ------ ----------
               PROFIS-II


  THE FRENCH REVOLUTION PROVIDED THE POLITICAL EVENT
WHICH GAVE SOCIOLOGY ITS START.  THROUGHOUT EUROPE AND
IN ENGLAND, THE QUESTION WAS ASKED:  WHY WOULD A PEACEFUL
SOCIETY SUDDENLY ERUPT IN VIOLENCE WHICH THEN CONSUMED
EVEN THE LEADERS OF THE REVOLUTION ITSELF?  WHAT WAS
IT THAT HELD PEOPLE TOGETHER IN SOCIETY ANYWAY?  DO
PEOPLE BASICALLY COOPERATE OR IS COMPETITION, FORCE OR
POWER THE KEY ELEMENT IN LIFE? VIOLENCE AND FEAR OF
VIOLENT CRIME IS STILL TODAY A GREAT CONCERN.  WHO
COMMITS CRIME AND WHY?

  THE SYSTEMATIC STUDY OF CRIME IS QUITE RECENT BECAUSE
BEFORE THE 1930S NO UNIFORM DATA WAS COLLECTED EVEN ON
SERIOUS CRIMES SUCH AS MURDER.  ONLY IN SELECTED CITIES
SUCH AS BOSTON DO WE HAVE RELIABLE DATA EVEN GOING BACK
100 YEARS.  FOR CRIME TODAY, HOWEVER, SYSTEMATIC STUDY
BASED ON FBI REPORTS AND SURVEY DATA IS POSSIBLE.

   DATA ON CRIME IS USUALLY PRESENTED AS A RATE.  THE
BASE USED FOR REPORTING CRIME RATES IS 100,000.

   THIS MEANS THAT A CRIME RATE OF 100 FOR A CITY WOULD
MEAN THAT 100 CITIZENS OUT OF EVERY 100,000 WOULD BE A
VICTIM OF A CRIME IN A YEARS PERIOD.

   NOW LET US CONSIDER SOME LEVELS OF CRIME IN AMERICAN
SOCIETY.  IN THE MID 1970S, TAKE A GUESS AT THE NUMBER OF
MURDERS PER YEAR PER 100,000 RESIDENTS OF A LARGE CITY
LIKE BOSTON.  ENTER ONLY A NUMBER (SUCH AS 513, AN
ABSURDLY HIGH GUESS) AND PUSH RETURN.

 MY GUESS FOR BOSTONS MURDER RATE IS;
!300

YOU ARE WAY TOO HIGH.  TRY AGAIN.
 MY GUESS FOR BOSTONS MURDER RATE IS;
!100

YOU ARE WAY TOO HIGH.  TRY AGAIN.
 MY GUESS FOR BOSTONS MURDER RATE IS;
!50

YOU MUST NOT LIKE BOSTON VERY MUCH.

  ACTUALLY IN 1973 IN BOSTON ONLY 5.7 PERSONS PER
100,000 WERE MURDERED.

   NOW, WHAT ABOUT THE PRESENT AS COMPARED TO THE PAST?
WERE THINGS WILDER IN THE PAST THAN AT THE PRESENT?
OR, WAS CITY LIFE SAFER IN 1850?
```

```
   ENTER WHAT YOU THINK THE MURDER RATE FOR BOSTON
 MIGHT HAVE BEEN BACK IN 1850:
!44
 YOU MUST BE A BELIEVER IN THE BAD OLD DAYS.

   BOSTONS LOWEST MURDER RATES WERE IN THE 1930S, WHEN
 THEY FELL TO A LOW OF ABOUT 1.5.  IN 1850 BOSTONS
 MURDER RATE WAS NEARLY 7.5, THE HIGHEST RECORDED.


          SOCIAL INDICATORS OF CRIME

   DID YOU EVER NOTICE ON THE BOTTOM OF RECORD CLUB
  OFFERS AND OTHER MAIL ORDER COUPONS THAT THEY ASK IF YOU
 HAVE A TELEPHONE.  THEY OFTEN DO NOT ASK FOR THE NUMBER--
 ONLY IF YOU HAVE A TELEPHONE OR NOT.  EVER WONDER WHY?

   BUSINESSES ARE INTERESTED IN SENDING PRODUCTS TO
 PEOPLE WHO WILL PAY FOR THEM.  RECORD CLUBS EXTEND A
 TYPE OF CREDIT, SINCE YOU ARE OBLIGATED TO BUY MORE
 RECORDS.  HAVING A TELEPHONE IS AN INDICATOR FOR A
 MANAGER THAT YOU HAVE MORE THAN A MINIMAL INCOME.
 BUSINESS MANAGERS HAVE FOUND THAT OWNERSHIP OF A
 TELEPHONE IS A GOOD PREDICTOR OF HOW WELL YOU WILL PAY
 YOUR BILLS.  IT IS OFTEN USED IN CONJUNCTION WITH:
 1. YOUR ADDRESS--WHERE IT IS IN A CITY AND:
 2. HOW LONG YOU HAVE BEEN LIVING IN ONE PLACE--STABILITY.

    OVER THE YEARS IT HAS BEEN DISCOVERED TIME
 AND TIME AGAIN THAT CRIME ALSO OCCURS MOST OF THE TIME
 IN POVERTY AREAS.  MANY OF THE SAME INDICATORS THAT
 BUSINESSPERSONS USE ARE VERY USEFUL IN PREDICTING CRIME.
 TAKE AGGRAVATED ASSAULT (AA HEREAFTER) FOR EXAMPLE.
 AGGRAVATED ASSULT IS MORE COMMON IN POOR AREAS
 OF CITIES, AND IS A CRIME AGAINST THE PERSON RESULTING
 IN SERIOUS BODILY INJURY.  HOW COMMON IS AA?  WHAT DO
 YOU THINK THE AVERAGE PERSONS CHANCE OF BEING BEATEN UP
 THIS YEAR ARE?  FOR EXAMPLE, IF YOU THINK YOUR CHANCE IS
 IS 1 IN 500, TYPE IN 500 AND PUSH RETURN.

   IN THE CITIES OF VIRGINIA AND NORTH CAROLINA, IF I
 WERE AN AVERAGE CITIZEN, I THINK MY CHANCE OF BEING
 A VICTIM OF AGGRAVATED ASSAULT WOULD BE 1 IN  ----
!20

  ACTUALLY, YOUR CHANCE IS ONLY ONE IN ABOUT 3000
 JUST SLIGHTLY HIGHER THAN YOUR CHANCE OF DYING IN AN
 AUTOMOBILE ACCIDENT.

   BUT POOR PEOPLE STAND A MUCH GREATER CHANCE THAN
 JUST AVERAGE OF BEING A VICTIM (OR A PERPETRATOR) OF
 A CRIME.  AGAIN, A GOOD INDICATOR OF THE TYPE OF AREA
 WHERE AA WILL BE MOST COMMON CAN BE INDICATED BY THE
 LACK OF TELEPHONES--THE SAME INDICATOR BUSINESS USES.
 IN VIRGINIA AND NORTH CAROLINA, SOME CITIES HAVE ONLY
 3.6% OF THE TOTAL HOUSEHOLDS WITHOUT PHONES WHILE OTHERS
 HAVE UP TO 26.1% OF ALL HOUSEHOLDS WITHOUT PHONES.


RUN
```

```
   NOW---LET US CHECK OUR LOGIC:
     1. IN POOR CITIES, ASSAULT IS MORE COMMON.
     2. LACK OF TELEPHONES IS AN INDICATOR OF A POVERTY AREA.
     3. THEREFORE, THE HIGHER THE PROPORTION OF ALL HOUSEHOLDS
        WITHOUT TELEPHONES, THE ------- THE CRIME RATE.

  IF YOU THINK THE ANSWER IS HIGHER, TYPE 1
  IF YOU THINK THE ANSWER IS LOWER, TYPE 2.

!1

 CORRECT.  THE HIGHER THE PROPORTION OF TOTAL HOUSEHOLDS WITHOUT
 TELEPHONES, THE HIGHER THE EXPECTED CRIME RATE.

    TO SEE HOW THE LACK OF TELEPHONES IS ASSOCIATED WITH
 THE RISE IN RATE OF AGGRAVATED ASSAULTS, TYPE IN BELOW 4
 VALUES OF YOUR OWN CHOOSING BETWEEN 3.7
 AND 26.0 FOR PROPORTION OF HOUSEHOLDS WITHOUT PHONES.
 USE ONLY A NUMBER SUCH AS 3.7, WITHOUT THE % SIGN. THE COMPUTER
 WILL CALCULATE YOUR CRIME RATES FOR YOU.

RUN
 ENTER YOUR FOUR VALUES, ONE AT A TIME.  PUSH RETURN AFTER
  EACH ENTRY:
!25.0
!20.6
!5.2
!12.9


 PROPORTION OF HOUSEHOLDS  !   PREDICTED AA RATE PER
    WITHOUT TELEPHONES     !    100,000 POPULATION
 ------------------------  !  ----------------------
LOWEST               3.5                142.447
 YOUR VALUE          5.2                169.562
 YOUR VALUE         12.9                292.377
 YOUR VALUE         20.6                415.192
 YOUR VALUE         25                  485.372
HIGHEST             26.1                502.917
-----------------------------------------------------

RUN

    CONCLUSION:  POVERTY IS A VERY CLEAR CORRELATE OF
 CRIMES SUCH AS AGGRAVATED ASSAULT.

              SEVERAL COMPLICATIONS
              ---------------------

    CONSIDER THE FOLLOWING CASES OF TWO CITIES:

              AGGRAVATED ASSAULT    PERCENT OF HOUSEHOLDS NOT
                      RATE            HAVING TELEPHONES

    CITY A           300                    8%
    CITY B           300                   25%

   HOW CAN THIS BE?
  THE ANSWER LIES IN WHAT IS KNOWN IN ADVERTISING AS
DEMOGRAPHICS. OTHER VARIABLES THAN TELEPHONE USE CORRELATE
WITH CRIME.

  TAKE THE FOLLOWING COMPLETELY HYPOTHETICAL EXAMPLE FROM
TELEVISION.  PROPORTION OF TOTAL VIEWING AUDIENCE IS
USUALLY TAKEN AS A MEASURE OF HOW POPULAR A SHOW IS.  BUT
IS IT?  WHO WATCHES WHAT SHOW?  LOOK AT THIS SITUATION:

  TELEVISION PROGRAM !   AUDIENCE BY AGE
       WATCHED       !  AGE 20-34 ! AGE 35-70
  -------------------!------------!-------------
1. CHARLIES ANGELS   !    100%    !     0%
2. LAWRENCE WELK     !      0%    !   100%
  -------------------!------------!------------
TOTAL AUDIENCE SHARE       50%          50%


   BOTH SHOWS HAVE HALF THE AUDIENCE, BUT DISAGGRATION OF THE
TOTAL REVEALS VERY DIFFERENT PATTERNS.  MAKERS OF THAT
FAMOUS REMEDY FOR TIRED BLOOD ADVERTISE ON THE LAWRENCE
WELK SHOW, NOT CHARLIES ANGELS.

  ANOTHER MEASURE WHICH CORRELATES HIGHLY WITH CRIME
IS PERCAPITA CITY FINANCES (IN DOLLARS).  VERY POOR
CITIES, WITHOUT MUCH OF A TAX BASE, HAVE LOW REVENUES.
OUR LOGIC IS THEN AS FOLLOWS:

    1. THE RICHER THE COMMUNITY, THE LESS THE CHANCE OF CRIME.
    2. WEALTH CAN BE MEASURED BY PERCAPITA CITY FINANCES.
    3. THEREFORE, THE HIGHER THE PERCAPITA CITY FINANCE RATES,
         THE ------ THE AA RATE IS EXPECTED TO BE.

 IF YOU THINK THE ANSWER IS HIGHER, TYPE 1.
 IF YOU THINK THE ANSWER IS LOWER, TYPE 2.
!2
RIGHT.  THE HIGHER THE PERCAPITA CITY FINANCES, THE LOWER
THE EXPECTED CRIME RATE.

 THE PUZZLE OVER CITY A AND CITY B CAN BE RESOLVED AS FOLLOWS:

   AGGRAVATED ASSAULT  NO TELEPHONES   PERCAPITA CITY FINANCES
CITY A    300                8%                   $88
CITY B    300               24%                   $394

 HIGHER PERCAPITA CITY FINANCES THUS CANCLED THE EXPECTED
INCREASE IN AGGRAVATED ASSAULT.

  LET US NOW TRY OUT SOME TYPICAL VALUES FOR PERCAPITA CITY
FINANCES (HEREAFTER PCF) TO SEE HOW THEY MIGHT INFLUENCE THE
CRIME RATE.  THE COMPUTER WILL PICK A RATE FOR TELEPHONE USE.
PICK A VALUE FOR PCF BETWEEN 83 AND 346.
RUN
```

```
ENTER YOUR FOUR VALUES
!85
!279
!400
VALUE IS ABOVE 347, TRY AGAIN
!200
!1
VALUE IS BELOW 81, TRY AGAIN
!279

     PERCAPITA CITY    !   PREDICTED AA RATE PER
     FINANCES IN $     !     100,000 POPULATION
     --------------    !   --------------------
LOWEST           81              424.504
YOUR VALUE       85              421.246
YOUR VALUE      200              327.564
YOUR VALUE      279              263.209
YOUR VALUE      279              263.209
HIGHEST         347              207.815

THE PROPORTION OF HOUSEHOLDS WITHOUT TELEPHONES IS
   ASSUMED TO BE    15.6298

RUN

          DENSITY AS A SOCIAL INDICATOR
          -----------------------------

    SOME SOCIAL SCIENTISTS HAVE STUDIED ANIMALS AS A CLUE
 TO WHAT HAPPENS WHEN HUMANS CROWD INTO CITIES.  CALHOUN
 HAS DONE THIS AND DISCOVERED THAT HIS RATS BECOME QUITE
 STRANGE WHEN CROWDED.  WHAT ABOUT HUMANS?

    DENSITY OF POPULATION PER SQUARE MILE WITHIN VIRGINIA
 AND NORTH CAROLINA CITIES RANGES FROM A LOW OF ONLY 258
 PERSONS PER SQUARE MILE TO A HIGH OF 7546 PERSONS PER
 SQUARE MILE.  HOWEVER, NEARLY ALL CITIES HAVE A POPU-
 LATION DENSITY OF 2000 PER SQUARE MILE OR MORE. WITHIN
 THE RANGE OF 2000 TO 7546, PLEASE ENTER 4
 VALUES FROM WHICH WE CAN CALCULATE THE SLIGHT EFFECTS
 OF DENSITY ON THE AGGRAVATED ASSAULT RATE.

RUN
ENTER YOUR FOUR VALUES
!3000
!4500
!5750
!6999

 DENSITY--POPULATION   !  PREDICTED AA RATE PER
  PER SQUARE MILE      !   100,000 POPULATION
 -------------------   !  ---------------------
LOWEST          2000             323.272
YOUR VALUE      3000             329.942
YOUR VALUE      4500             339.947
YOUR VALUE      5750             348.284
YOUR VALUE      6999             356.615
HIGHEST         7546             360.264
----------------------------------------------
```

```
RUN

  AS YOU CAN SEE, THE RELATIONSHIP IS A VERY SLIGHT ONE.
  STATISTICS WOULD TELL US THAT IT IS INSIGNIFICANT.
  TAKE A GUESS AT THE SIZE OF THE CORRELATION BETWEEN
  DENSITY AND AGGRAVATED ASSAULT?
!.1

    ACTUALLY IT IS A TINY .036.  THE CORRELATE BEETWEEN
 SERIOUS CRIME AND DENSITY IN ALL AMERICAN CITIES IS ABOUT
 THE SAME.

  FOR THE 51 NATIONS OF THE WORLD FOR WHICH WE HAVE DATA,
 THE CORRELATE BETWEEN THE MURDER RATE AND POPULATION DENSITY
 IS -.09, NOT ONLY WEAK BUT IN THE WRONG DIRECTION.
 ACTUALLY DENSITY IS NOT A GOOD INDICATOR OF CRIME
 WITHIN CITIES OR BETWEEN NATIONS.

                    CONCLUSIONS
                    -----------

    WE HAVE SHOWN YOU HOW SEVERAL SOCIAL INDICATORS CORRELATE
 WITH THE AGGRAVATED ASSAULT RATE.  THE COMPUTER IS NOW
 PROGRAMMED TO ALLOW YOU TO CHANGE ALL 3 VARIABLES AS YOU
 WISH.  AFTER THREE TRIES, YOU MAY QUIT IF YOU WISH OR
 TRY TO FIND THE BEST VALUES FOR A PERFECT SOCIETY (HOWEVER
 YOU CHOOSE TO DEFINE IT.)

PICK A VALUE FOR PERCAPITA CITY FINANCES (RANGE:83-347):
!347
OK-PICK A VALUE BETWEEN 2000 AND 7546 FOR POPULATION DENSITY
!7050
NOW PUT IN YOUR VALUE FOR PROPORTION OF HOUSEHOLDS
WITHOUT PHONES (THE RANGE IS 3.5 TO 26.1):
!3.5


 YOUR CALCULATED AA RATE IS
**********
   114.615
**********
 OK, NOW LETS TRY A SECOND SET OF VALUES.
PICK A VALUE FOR PERCAPITA CITY FINANCES (RANGE:83-347):
!83
OK-PICK A VALUE BETWEEN 2000 AND 7546 FOR POPULATION DENSITY
!7546
NOW PUT IN YOUR VALUE FOR PROPORTION OF HOUSEHOLDS
WITHOUT PHONES (THE RANGE IS 3.5 TO 26.1):
!25


 YOUR CALCULATED AA RATE IS
**********
   687.762
**********
 NOW LETS TRY A THIRD SET:
PICK A VALUE FOR PERCAPITA CITY FINANCES (RANGE:83-347):
!
```

```
 347
OK-PICK A VALUE BETWEEN 2000 AND 7546 FOR POPULATION DENSITY
!7440
NOW PUT IN YOUR VALUE FOR PROPORTION OF HOUSEHOLDS
WITHOUT PHONES (THE RANGE IS 3.5 TO 26.1):
!8


 YOUR CALCULATED AA RATE IS
**********
   190.248
**********

TO TRY MORE VALUES, TYPE 1.  TO OBTAIN HOMWORK, TYPE 2.
!2


                 HOMEWORK
                 --------

   BELOW IS THE APPROXIMATE EQUATION USED TO PREDICT
 AGGRAVATED ASSAULT.  PLEASE SOLVE FOR THE VALUES
 SPECIFIED BELOW BY HAND SO YOU CAN GET SOME FEEL FOR
 THE PROCESS. YOU MAY DO THE CALCULATIONS MOST EASILY
 WITH A CALCULATOR.  WHEN DONE, TURN IN THIS ENTIRE
 SHEET FOR CREDIT.

AA = 209 + .03 (DENSITY) + 14.6 (PROPORTION WITHOUT
 TELEPHONES) - .9 (PERCAPITA CITY FINANCES).

       FILL IN VALUES ASKED FOR

  AA          PCF         NO TEL          DENSITY
  ?           $98          20%             2980
  ?          $300          20%             2980

  IF A CITY HAD RECORDED 88 ASSAULTS AND HAD A POPU-
  LATION OF 52,300, WHAT WOULD THI ASSAULT RATE BE?------
```

Footnotes

[1] Support for this paper was provided by grant #SER77-02431 from the National Science Foundation under the LOCI program.

[2] Teaching in a small college means that it is not possible to have large sections to use as a control group. The control group used here was that of another professor who allowed the use of his class as an experiment. All students involved in the experiment were women. Due to the small size of the samples, a significance level of .1 was used in Tables 2 and 3.

References

Conklin, George H. (1976) "Using the Computer in Introductory Sociology." Teaching Sociology 4 (October):83-95.

(1977) "Experimental Simulation in Introductory Sociology." Proceedings of the 1977 Conference on Computers in the Undergraduate Curricula CCUC8:313-322.

Davis, James (1978) "Social Facts, Yes!" Forthcoming in Teaching Sociology.

Lorber, M. A. and S. J. Skinner (1977) "Computers and Teacher-Education." Proceedings of the 1977 Conference on Computers in the Undergraduate Curricula CCUC8:87-91.

DEVELOPMENT OF COMPUTER-LINKED EDUCATION IN RESEARCH PSYCHOLOGY

John L. Horn and J. Jack McArdle

University of Denver

The purpose of this paper is to report on the development and status of an on-going research project at the University of Denver, funded for three years by a large grant from the National Science Foundation's Research Initiation and Support (RIAS) agency. We hope that the implications and results of this effort will be fruitful for undergraduate and graduate education in all of the social sciences, and we welcome such discussion.

Planning/Assessment

1. Broad Assessment of Needs

A recent report from the National Board on Graduate Education (1975) states that needed for the future are scientists trained in new molds, capable of applying their skills and knowledge in entirely new ways, able to take jobs and do research in settings not heretofore regarded as fertile ground for scientific investigation, and willing to tackle problems which traditionally have not been properly addressed with the methods of science. This call for a new breed of scientist is perhaps best directed at the social sciences, perhaps particularly psychology. The present proposal is an effort to heed this call by structuring experimentation/statistical/computer training in a mold that many will recognize as much needed not only in psychology, but also in the behavioral sciences generally, and perhaps in other sciences as well.

In many subareas of psychology it is clear that the field has not yet committed itself to any conception of human nature that could apply beyond the confines of the laboratory. The discipline is moving towards reformulations that more closely link the laboratory to the *in situ* conditions in which people live. This does not mean an end to laboratory experiments in the future, but a commitment to the study of variables that are ecologically important, rather than those that are easily managed.

Another factor of importance in forecasting the future for psychology is technological advance for obtaining comprehensive and refined sets of behavioral data in scientific experimentation. For example, contingency data, in which presentation of stimuli are made contingent on particular patterns of behavior,

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

can now be rather easily gathered. The future will bring many such changes in the data base of behavioral science. Psychologists will need to meet the challenge of making effective use of such data.

The psychologist of the future must be able to bring the best of the experimental expertise developed in the laboratory to the community. Also, these scientists must have the in-depth understanding of experimental design, measurement, and mathematical-statistical methods that will enable them to creatively and effectively deal with emerging problems occasioned by gathering data in unusual ways and under unusual (e.g., naturalistic) conditions.

a. Computerized Experimentation

One of the major technological advances of the last decade has been the development of computers to help execute and control psychological experiments. The use of the computer for on-line control of experiments has revolutionized research in several ways. The computer can assist at all stages of experimentation, from the generation and randomization of stimuli, through the running of subjects, to the mathematical-statistical analyses of the results. Experiments that now require months for the gathering of data by traditional methods can be run in a few days using a computer, and the results can be analyzed within minutes after the completion of the experiment.

Until recently, however, the advantages of computerized research were accompanied by several severe problems. The high cost of the system and the technical expertise required put such systems out of reach of all but a handful of scientists. This has meant that very few of the existing cadre of behavioral scientists, now training the scientists of tomorrow, are sufficiently familiar with the capabilities of computerized research to be able to teach this to their charges. Thus tremendous potential has remained largely undeveloped.

A second major problem of the past was that the size of the machines required for computerized research made portability out of the question. Use of computers thus served to tie researchers to laboratory settings and to make research fit the laboratory rather than fit *in situ* questions about human functioning.

In the last few years, however. advances in the computer industry have largely eliminated these problems. The cost of a small computerized laboratory has dropped dramatically and an equally dramatic drop in the size of the machines. It is now possible to carry one's computerized laboratory into the field to do research. The recent developments of hardware and software packages greatly reduce the technical expertise needed. With only a relatively small amount of training, the researcher can now maintain and run a computerized laboratory with a minimum of outside help. Unfortunately, this small amount of training usually is not available to the aspiring researcher in the behavioral sciences. But such training can be, and should be, a basic part of forward-looking programs.

b. Statistical/Mathematical/Measurement Skills

Behavioral scientists are becoming increasingly aware that measurement and statistical procedures used most frequently in the past often were not, even then, maximally appropriate for the data. The inadequacies of these procedures are often seen most clearly when research moves from the laboratory to the community (as documented in recent critiques, as by, for example, Horn, 1970; Horn & Donaldson, 1976), but the lessons are there to be learned in other ways.

Increasingly, it is being recognized that the major concepts of psychological theories require multivariate measurement models and the means for analyzing many independent and dependent variables (Cattell, 1966). Increasingly, too, psychologists are discovering that the data that are most relevant to evaluation of major theoretical positions can be gathered only with quasi-experimental designs (Campbell & Stanley, 1963). Relatedly, there is increased need for behavioral scientists who can do exploratory data analyses and use Monte Carlo methods when these are needed. Most statistical training in the behavioral sciences has focused almost exclusively on confirmatory data analysis. But as Tukey (1976) has pointed out, "Unless exploratory data analysis uncovers indications, usually quantitative ones, there is likely to be nothing very interesting for confirmatory analysis to consider . . . restricting one's self to the planned analysis--failing frequently to be comfortable." With the computer capabilities now available and an awareness of the uses and abuses of Monte Carlo procedures, such "messy data" often can be analyzed in ways that provide useful (albeit rough approximate) answers to important scientific problems.

To deal with the problems of contingent dependencies and near-continuous observations, to handle multivariate measurements gathered with quasi-experimental designs, to explore creatively but with intelligent caution, to "unlock the analysis of a body of data, to find the good ways to approach it" (Tukey, 1976), requires scientists who have a good understanding of what can and cannot be accomplished with the many existing methods innovatively and, most important, they must be ready to go beyond particular methods and look at data in new ways. As psychology moves towards more ecologically valid models and experimentation, there will be increased need for this kind of behavioral scientist.

Much of the capability for meeting the need to produce substantial numbers of young scientists in this mold is at our fingertips, thanks again to advances in computer hardware and software. Until recently the behavioral scientist who became adept in data analysis worked long hours in acquiring programming skills. The great majority of others became dependent on canned programs, which eventually dictated data-gathering design. The problems can, in principle, be overcome by new software-hardware, such as the APL language, and by multipurpose software packages, such as P-Stat (Buhler, 1976). These capabilities make it possible to develop the kind of scientist-cum-data-analyzer described briefly in previous paragraphs.

In sum, then, the need for a new kind of behavioral scientist is pressing, and the means for producing substantial numbers of this new professional are at hand. Needed are enterprising programs designed to indicate how the training of such new professionals should best proceed. The present paper is an outline of such a program, currently running at the University of Denver.

2. Project Objectives

In very broad terms, the plan is to improve facilities for collecting and analyzing data and to provide training for both faculty and graduate students in the use of these facilities. Alternative approaches to computerized control of research are to be explored. These alternatives are to be evaluated in order to determine which are most cost-effective, most easily taught, and easiest to use. The capacity of the university computing system is to be expanded to include several of the currently most powerful procedures for analysis of behavioral science data. A training program is to be developed to enable researchers to make full use of these capabilities. Workbooks and reports of follow-up evaluations are to be provided to help other disciplines and other institutions to profit from this program development.

A major objective of the training program is to produce psychologists who can, without dependence on programmers, make well-informed data-gathering and data-analysis decisions in which substantive-scientific issues are foremost (statistics serve rather than dictate research). A second and related major objective is to teach the young scientist how to most effectively use the computer (i) to gather responses and monitor experiments, (ii) to construct models for

190

behavior and compare these with actual, emerging patterns of behavior, and (iii) to test the adequacy of fit of various possible models. These two major thrusts of the program are to be melded together, so the researcher will be able to make effective use of the computer, and related capabilities, at all phases of experimentation, from initial observation to final table of results. And throughout the training there is to be emphasis on experimentation that is relevant to understanding the on-going functioning of the human in his natural environment.

The program of this proposal will most directly strengthen the psychology department capabilities for educating young scientists, but as parts of the program are developed and become fully operational, they will be incorporated into the training programs of other departments, particularly mathematics (i.e., the statistical program), sociology, and education. There is strong and general support for the project within the university.

Proposed Plan/Strategies and Activities

1. Overview of the Plan

We believe that the overall goal of training researchers to be self-sufficient in the many uses of modern computer technology can be attained most efficiently by providing extensive opportunities for hands-on use of computer capability in the context of the kind of experimentation and data-analyses that emerge from considerations of the content and issues of a discipline. A laboratory/classroom (Labclas) will be produced to help realize both the instructional and the conduct-of-research objectives of the program. When used as a classroom, the Labclas will enable the behavioral science instructor to conduct laboratory demonstrations analogous to those conducted in the training of such scientists as chemists and physicists. The facilities of the Labclas will permit the instructor to formulate complex questions which, however, can be answered quickly by students who will be using the computer via terminals. Students will be taught to run themselves as subjects, to construct models for human functioning, and to examine the plausibility of these models with patterns of their own behavior. The Labclas will be used to help the young scientist to learn both data-analysis procedures and experimentation in terms of hypothesis testing and exploration of alternatives, rather than in terms of cut-and-dried textbook conclusions. When used as a laboratory, the Labclas will be a computing and experimentation center. Some of the data-gathering components of the laboratory will be portable, so the researcher may carry his experimentation into the field.

2. Establishing the Labclass and Field Experimentation Capabilities

a. The Labclas and Remote Unit

The laboratory/classroom will be equipped with six APL transmitter-receiver terminals, each having CRT and hard-copy capability. The terminals will also be equipped with a polytronics reaction time collection device. This will permit highly accurate reaction times to be collected externally from the time-shared computing facility.

A single terminal unit will also be located at the Child Study Center, a building that is about one-fourth mile away from the main set of psychology department buildings (where the Labclas is to be located). It is important to have this remote unit, because many of the students and faculty in clinical psychology are located at this center, and the faculty of this branch of the department are active researchers who will be very much a part of the effort to improve research in accordance with the objectives of this proposal.

Telephone hookups will provide terminal access to the university computing center's Burroughs 6700, a third generation computer, and a new PDP 11/70 computer. A switch will enable the user to select which of the two possible computers to access at a particular time. Tape, disk and microprocessor storage components will permit the user to move output from one computer to another in order to fully exploit the unique capabilities of each machine.

b. The Field Laboratory (Fieldlab)

The terminals of the Labclas and remote unit are detachable and portable. Thus in any field research situation in which telephone hookups may be made, one or more of the terminals may be used to bring the capacities of the PDP 11/70 or B6700 computers to experimentation. To further enhance this capability for carrying the laboratory into the community, even when telephone hookups are not possible, two PDP LSI-11 microprocessors, each with 4k memory and serial interface, will be obtained. To make it possible to run perception and cognition experiments involving graphics and dynamic displays or tachistoscopic presentations, it is necessary to have a digital-to-analogue converter interface to drive the department's point-plot scope. For accurate timing of various parts of experiments it is necessary to have a programmable clock. In terms of cost and the objective of achieving portability, by far the most important item is a floppy disk system with a large number of spare disk cartridges. This provides the storage capacity needed to present extended arrays of stimuli and collect corresponding arrays of responses.

c. The Addition of a PDP 11/70 Computer

To permit rapid on-line processing for the Labclas terminals and access for the minicomputers, additional computer power is required. This will be handled by the acquisition of a PDP 11/70 computer to serve as a front-line to the B6700 main system. The integration of these two systems will allow a multiuser mix for many large on-line tasks, such as APL.

d. These Facilities in Relation to the Project Objectives

With these additions to the existing university computing systems, the basis for a very powerful and flexible experimentation/data analysis system will have been established. This system will provide the foundation for constructing an innovative program of training in the psychology department. It will also provide a basis for enhancing the training of scientists in other departments of the university.

3. Building the Training Program

The process of introducing computer-facilitated instruction into the undergraduate training program is to be accomplished in a step-wise manner, beginning with courses in measurement, statistics, and cognitive processing, first at the graduate level, then at the undergraduate level. The plan is to acquire, create, and try out a considerable number of computer lessons for each of several areas of psychological research and put these in a set of workbooks from which an instructor may select those lessons that seem to be appropriate for a particular course. These books will be made available to individuals who are interested in establishing their own computer-based educational programs.

The following sections provide examples of the way these training modules are to be built. There will be laboratory demonstrations, self-directed and self-initiated experimentation and analyses, group projects, lecture-discussions, field trips, etc. A short example will be described here.

a. Teaching Awareness of Experimental Paradigms and Models

Much of a psychology student's course work involves exposure to the methods and results of a large number of experiments. Unfortunately, often the outcome of this exposure is a student who carries away some facts but little understanding of the methods used to obtain these facts. For this reason the aspiring researcher often lacks basic information needed to critically evaluate the facts and to gain insight into the assumptions and manner of generating scientific knowledge. While this problem has long been recognized, there has been in the past no ready solution. But the capabilities to be established in this project provide a basis for dealing with these problems. Research experience by way of computer simulation permits an instructor to arrange for concrete involvement with many experiments under conditions that closely approximate those in which the studies were originally run. The module consists of a series of experiments in cognitive psychology which have been programmed (Bewley, 1974) to run in BASIC. The program allows the student to run himself as a subject in the various experiments while the computer collects the data. The simulation is based on a particular model for problem solving. In running the simulation, the student can see how the model is selecting responses at each step of the problem, and can compare his performance with that of the model as a means of assessing the validity of the model. The student may easily run several versions of any given experiment by manipulating independent variables, thus enabling him to assess the effects of these variables on his own behavior as well as that of the model. The program permits him to run large numbers of simulated "subjects" for each experiment in a matter of a few minutes. An important consequence of this kind of instruction is that students, once exposed to a large number of computer models, become interested in constructing their own models.

In building the training program of this proposal, modules of the kind outlined in the example are to be assembled for a considerable array of simulated experiments in several major areas of psychological research. These are some of the new "lessons" to be incorporated into the existing graduate training program and to be made available for other institutions and departments. Such lessons already available include one in interactive decision-making developed for social psychology and several for perception courses.

b. Teaching Data Analysis

Modules of the kind described above are to be developed for training in psychological measurement and mathematical-statistical analysis. In addition, for this part of the educational program, some of the most powerful and up-to-date programs for use in data analysis are to be incorporated into the university computing system. Training in both theory and applications is to be built around an integrated set of these programs.

To accomplish the ultimate objectives in this regard, a series of workshops will be conducted to provide guidelines for curriculum development. The workshops will serve several functions. Principally, however, they will enable instructors to develop classroom techniques for utilizing the computer in both exploratory and confirmatory data analyses. Although a part of this endeavor will be a lecture series by well known scientists, the workshops are intended to focus in on a single set of techniques for an intensive five day period run by an expert scientist. Workshops will be developed by first obtaining relevant statistical programs and making these operational on our computer configuration. Next we will run a training session on the specific techniques taught by a well-known expert in the field. These workshops tentatively include:

1. Bayesian Statistics: Melvin Novick on the CADA (1975) system.
2. Exploratory Data Analysis: John Tukey on the IEDA program and EDA philosophy.
3. Monte Carlo Methods: Frank Baker on general Monte Carlo techniques.

4. Time Series Analysis: Gene Glass on the use of Time Series in Developmental Psychology.

A number of statistical software packages are currently being evaluated for general inclusion into the computing environment. A tentative list follows:

1. EFAP, COFAMM, and LISREL--Jöreskog.
2. MULTIVARIANCE IV--Finn
3. MULTIQUAL--Bock
4. RUMMAGE--Scott
5. P-STAT--Buhler
6. Omnitab II--Hogben, Peavy, and Varner.

The integrated instructional program to be built around a selected set (and selected parts) of these various computing packages will go a long way toward the objective of freeing young scientists from dependence on programmers and statistical consultants.

In addition to developing the above-mentioned software capabilities, the instructional program will develop an ability to use APL. Iverson's APL is currently probably the best available language for enabling a person who knows the mathematics of his analysis to do these analyses directly on the computer, without having to learn detailed programming skills. APL is an exemplar of the kind of computer language that is "for tomorrow," in the sense that very likely much of the basic thinking of scientific analyses using the computer will be done in languages similar to APL. By teaching APL capability to the aspiring scientist, the program can help to prepare him to use the computer to help think through a variety of possible models and procedures for data analysis before actually setting out to "crunch the data." Also, of course, APL is presently the language, par excellence, which enables the teacher to develop the theory of mathematical-statistical-mensuration methods and model testing in close contact with actual computing and evaluation of results.

4. Implementation and Progress

a. The Three Year Plan

During the first year the necessary physical transformations will be undertaken to create the new computer configuration and the Labclas. Development and research on the lessons and modules will begin during this time so we will be ready to go when the physical configuration is completed. The necessary paper work involved in setting up the workshop sessions, obtaining the Fieldlab equipment, and obtaining the necessary statistical software will also begin. The very notions about the general use of the computer in psychology will be entertained in virtually every current class in psychology, and special general computer/psychology lectures will be given. In short, we will begin to get ready for heavy schedule of the next few years.

An important part of this project is the joint effort by faculty members of several departments on campus. This is especially important during the first year. Both the mathematics department and the computer center have helped a great deal in the formation of these ideas and their expertise (see Dorn, 1968; Rechard & Ives, 1973) will be actively sought. The psychology department itself is staffed by several well-known people in the field of measurement/statistics (see Gollob, 1968; Horn, 1971) and computer/cognition (see Keenan, in press; Potts, 1976). In fact, this group represents the major decision-making body for the project and will take full responsibility for project direction.

During the subsequent years we will carry out the plan proposed here. No doubt this will require some reorganization and rethinking. Heavy emphasis will be placed on evaluating the project in terms of other programs in psychology. A final document describing the full efforts and outcomes of this experience will be available at the end of the third year of operation. The project itself will continue, hopefully in the multidisciplinary fashion in which it is intended.

b. Progress

As of this date (January 1978) we are moving ahead at a very rapid pace. Construction on the Labclas has begun, and we have obtained much in the way of computer hardware and software.

In a joint effort with the computer center, we have purchased a PDP 11/70 (120K). The Labclas is being set up to have six Data-Media 1500 CRT/APL terminals and a printer on a wheel base for ease of transport around the Labclas. These have high speed modum connections to both the B6700 and the PDP 11/70. Until such time as two systems are software compatible, a separate magnetic tape drive unit will allow for data communication. The purchase of two Texas Instruments Silent 743 portable terminals allows flexible off-campus programming. The portable microprocessor is already in use by several research studies both on and off campus.

The acquisition of statistical software has begun by already obtaining almost all programs mentioned (except for RUMMAGE and OMNITAB). As such, we are now pursuing several other programs and packages. In lieu of workshops without the Labclas we are concentrating on guest lectures. In this regard we have had two excellent sessions by Dr. David R. Heise (Department of Sociology, University of North Carolina) on "Causal Models in the Social Sciences" and Dr. Herman A. Wold (Department of Statistics, University of Göteborg, Sweden) on "Soft Models--Intermediate Between Statistics and Data Analysis." Additionally, a four-quarter sequence in the logic and design of multivariate models has begun as part of the regular modular series.

In summary, progress is rapid although we are in the necessary preparatory waiting period for the physical structure of the Labclas. The expected completion of the Labclas facilities is March 1978. By this time we hope to be in full swing.

5. Final Comment

Although we have engaged many experts on and off campus, we actively seek counsel with those who can appreciate the efforts we place in front of this mammoth challenge. We will be very grateful to receive your thoughts and ideas about this project, now or in the future.

References

Bewley, W. L. Cognitive Psychology: A Computer-oriented Laboratory Manual. Hanover, N.H.: Project COMPUTE, 1974.

Buhler, R. The P-Stat System Status in March 1976. Paper presented at Computer Science and Statistics: Ninth Annual Symposium on the Interface, April 1976, Harvard University.

Campbell, D. T., & Stanley, J. C. Experimental and quasi-experimental designs for research on teaching. In Gage Handbook for Research on Teaching. Chicago, Ill.: Rand McNally, 1963.

Cattell, R. B. (Ed.). Handbook of Multivariate Experimental Psychology. Chicago: Rand McNally, 1966.

Dorn, W. S. The computer: A mathematics laboratory. Journal Education Data Processing, Spring 1968, 5, No. 2, 67-80.

Gollob, H. F. A statistical model which combines features of factor analytic and analysis of variance techniques. Psychometrika, 1968, 33, 73-115.

Horn, J. L. Review of J. P. Guilford's "The Nature of Human Intelligence," Psychometrika, 1970, 35, 273-277.

Horn, J. L. Integration of concepts of reliability and standard error of measurement. Educational and Psychological Measurement, 1971, 31, 57-74.

Horn, J. L., & Donaldson, G. On the myth of intellectual decline in adulthood. American Psychologist, 1976, 31, 701-719.

Keenan, J. M. Processes in classifying pictures and words: Parallel? Independent? Obligatory? In J. W. Cotton & R. L. Klatzky (Eds.), Semantic Factors in Cognition. Hillsdale, N.J.: Lawrence Erlbaum Associates (in press).

Novick, M. R. A course in Bayesian statistics. American Statistician, 1975.

Potts, G. R. Use of a campus-wide timesharing computer to run reaction time experiments. Behavior Research Methods and Instrumentation, 1976, 8, 179-181.

Rechard, W. O., & Ives, F. A mini-computer network for interactive data management in a large scale data processing environment. International Conference on Communications, Conference Record, June 1973, Vol. II, I.E.E.E.

Tukey, J. Exploratory Data Analysis. New York: Academic Press, 1976.

NOTE

This report and project is funded by NSF Grant #SER77-06935 under the same title.

QuATE

A STANDARDIZED PROCEDURE FOR DESCRIBING STRUCTURED CAI MATERIALS

by

Doug Aims

Merced College

INTRODUCTION

Computer Assisted Instruction (CAI) needs to be made understandable to teachers from all disciplines. It is not enough to demonstrate what is possible using "state-of-the-art" techniques and the most sophisticated equipment available. Teachers need a simple way of "getting started" so they can discover for themselves the possibilities and benefits of CAI. They need "hands-on" experiences and opportunities to share these experiences with others who are "talking their language." That is the purpose of the QuATE procedure which is described in this paper.

QuATE is a standardized procedure for describing structured CAI materials. Anyone who learns the procedure should be able to write, and perhaps publish, machine-independent CAI materials which can be easily read and understood by others who have learned the procedure. As you will see after reading this paper, one of QuATE's greatest strengths is simplicity. The procedure is both easy-to-learn and easy-to-use. Another important strength is its stability. Changes to QuATE will be resisted. Transportability is a third important advantage. The procedure can be implemented on a variety of computers ranging from large time-sharing systems to small stand-alone microcomputers. A QuATE processing program can be written in a high-level language such as BASIC.

The steps associated with the procedure are partly indicated by the letters in the name QuATE. A question is presented and the user responds; this response is compared to a stored answer; if the user response matches the stored answer, then a reply is followed by an appropriate "branching action"; else (if there is no match) a different reply and an appropriate "branching action" result. Figure 1 illustrates this logic.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

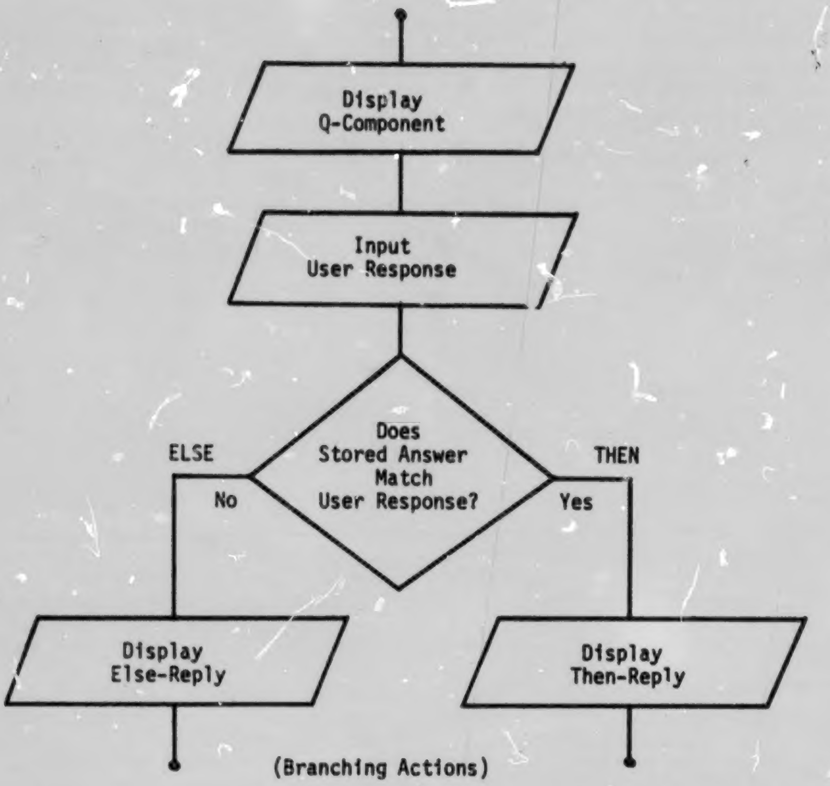


Figure 1. QuATE Logic Structure

QuATE ITEMS

Each CAI unit consists of a sequenced set of items. A QuATE item consists of the following four components: (1) Q - Question; (2) A - Answer; (3) T - Then reply; and (4) E - Else reply. These components are written by the author of the unit and then stored in the computer's memory. The Q-A-T-E order is important. An optional form which requires only the Q and A components may be used; in this case, then and else replies are supplied by the QuATE processing program.

The Q-component may contain comments instead of a question, or they may precede the question. Comments might be used to explain or clarify something the student is expected to know. They might also direct the student to manually operate an audio-visual device (e.g., a tape recorder or slide projector); or look at a page in a book or printed handout; or interrupt the processing program and restart at some specified point in the unit. In any case, an actual question does not have to be displayed on the computer terminal as long as something is displayed which prompts the user to make a response.

Every Q-component must be followed by an A-component. The A-component may be an alphanumeric, numeric, or percentage answer line. Alphanumeric answer lines are used when the user response is expected to match the stored answer exactly (i.e., character-for-character). Numeric answer lines allow a range of accuracy to be associated with each answer; therefore, an exact match is not necessary.

Furthermore, equivalent forms of the same answer (e.g., 100 and 100.0) cause no problems when numeric answer lines are used. Percentage answer lines allow a pass/fail criterion score to be associated with the answer. After commas and blanks are removed, the response and answer key are compared character-for-character. An exact match is not necessary.

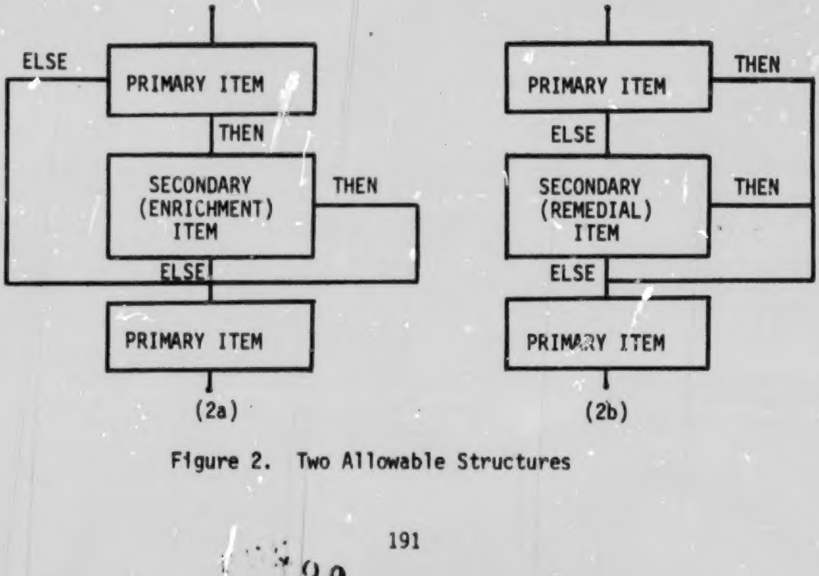
The T-component and E-component are similar. They enable the computer to supply the student with feedback in the form of comments or instructions. For example, if the user response matches the stored answer, the computer could display the following then-reply: YES, THAT IS CORRECT. On the other hand, if the user response does not match the stored answer, the following else-reply could be displayed: NO, THE CORRECT ANSWER IS ANTARCTICA. Branching actions are also indicated by these components; however, the possible actions will be discussed later.

Line Codes. As mentioned before, item components are identified by the letters Q, A, T, and E. One of these letters must appear in the first position of each line. The code in the second position further identifies the line type. Ten codes are allowed -- C, P, S, @, #, %, J, N, R, and H. The third position of each line contains a colon (e.g., QP:, A#:, TJ:, ER:).

[QC:, TC:, and EC:] Only one answer line is allowed per item; however, multiple Q, T, and E lines are allowed. The letter C is used in the second position to denote a continuation line. Any number of continuation lines may be used; the QuATE processing program pauses periodically to allow the user to read the information before it "rolls off" the display screen.

[QP: and QS:] An item may be a primary item or a secondary item. Primary items are presented to all students. Secondary items are provided for remedial or enrichment purposes and are only presented to some students. Figure 2 shows two allowable structures, each having two primary items and one secondary item. In the first case (2a) the secondary item is displayed when the preceding primary item results in a matched response. In the second case (2b), the secondary item serves a remedial purpose since it is received when the preceding primary item does not produce a matched response.

Primary items are denoted with the letter P in the second position of the initial Q-line; a letter S in the second position denotes a secondary item. In all other respects, primary and secondary items are described in the same way.



(2a) (2b)

Figure 2. Two Allowable Structures

[A@:, A#:, and A%:] Alphanumeric answers are identified by entering the special character @ (at-sign) in the second position of the A-line; numeric answers by the special character # (pound sign); and percentage answers (e.g., multiple-choice test answer keys) by the special character % (percent sign). If a numeric value is represented on an alphanumeric answer line, the response will have to match exactly since the comparison is made character-by-character. Therefore, 100.0 would not match 100. When an alphanumeric string of characters appears on a percentage line, the response does not have to match exactly because a pass/fail criterion is used.

[TJ:, TN:, TR:, TH:, EJ:, EN:, ER:, and EH:] The first T and E lines in each item may specify an appropriate branching action. Four actions are possible; the same four are allowed for each type of line. These actions and their mnemonic code letters are listed below:

1. J - JUMP to the next primary item
2. N - Continue with the NEXT item (primary or secondary)
3. R - REPEAT the current item
4. H - HALT

A letter (J, N, R, or H) is entered in the second position of the line to specify the desired branching action.

Since each item has two components which allow branch specifications (T and E), sixteen branching combinations are possible. Only a few of these combinations need to be used. Among the commonly-used ones are the following: TJ/EN, TJ/ER, TN/EJ, TJ/EJ, and TH/ER. The TH/EN variation is used when a secondary remedial item is to follow. One of the most useful variations is the TJ/ER item; it allows the user to "try again." If an enrichment item is to follow, the TN/EJ variation should be used. Finally, the TH/ER form is useful as the last item in a CAI unit.

When the T and E components are omitted, the QuATE processing program supplies TN:RIGHT! as the T-component and ER:WRONG! as the E-component. This type of item is referred to as a Q/A item.

The repeat code, R, allows the user to try again; however, it does not cause the Q-component to be repeated. Each time an "incorrect" response is given, the else-reply is printed followed by a message directing the user to "try again." When the user has responded three times to the same item, the QuATE processing program prints out the "correct" answer and then continues with the next item.

[LL:] The last line in a CAI unit is denoted by entering LL: in the first three positions of the line. Any message may appear on the line following the colon. When the LL: is detected during processing, the message is printed and processing halts.

Questions, Then-Replies, and Else-Replies. Starting at position four, each Q, T, and E line may contain capital letters [A - Z], decimal digits [0 - 9], blanks [ ], and 21 common special characters [+-*/,.$'()?=#<>!%&@:;]. The length of the line, including the first three positions, is limited to 64 characters. Even though longer lines and additional special characters might be desirable, the limitations enhance transportability of QuATE materials.

Answers. Answer lines have the following possible forms:

1. A@: character string
2. A@: character string; character string
3. A#: decimal value
4. A#: decimal value; range value
5. A%: character string
6. A%: character string; criterion (pass/fail) score

Character strings in answer lines may contain the same characters allowed in Q, T, and E lines with one exception; the semi-colon is not allowed because it is used as the separator in forms 2, 4, and 6 above. If all positions after the colon are blank, the item is treated as a comment item; processing immediately continues with the next item in the unit.

Alphanumeric answer lines allow one or two character strings to be specified as answers. The second string is used when either of two responses is considered a matching response. A match occurs when the response is the same as either stored answer. Comparison starts with the first non-blank character in each string.

Numeric answer lines may contain an optional range value. If no range is specified, the QuATE processing program supplies a range value equal to 0.01 percent of the correct answer. For a match to occur, the response must be equal to the decimal value plus or minus the range value. Commas and blanks are ignored; therefore, 30,000 is equivalent to 30000.

Numeric values may be expressed in exponential form. This form uses the letter E to mean "times ten to the power of." A value such as 1,500,000 could be entered as 1.5E6 or 15E5. The form is useful for expressing very large and very small numbers. It is often referred to as scientific notation.

Percentage answer lines may contain an optional pass/fail criterion score. If no criterion score is specified, the QuATE processing program uses 70 percent of the string length as the criterion score. The string length is determined after commas and blanks have been removed. After the answer string and the user response are compressed, the strings are compared character-for-character. If the number of matching characters is greater than or equal to the pass/fail criterion score, the then-reply and associated branching action are used.

WRITING QuATE UNITS - SOME SUGGESTIONS

1. Do not write long units. Short units that can be completed in approximately 15 minutes are recommended.

2. If only one alphanumeric answer is stored for an item, write the question in such a way that only one correct response is possible. If a second (optional) alphanumeric answer is stored, make sure only two correct responses are possible. Extra information should be provided to limit the correct responses.

3. Use printed handouts freely. It is not necessary, or even desirable, to display everything on the computer terminal. Printed handouts are particularly effective as a means of displaying graphics, criterion-referenced tests, text material, and formulas.

4. Select two or three branching combinations and use these to write most of the items in each unit. TJ/ER and TJ/EN items are likely to very popular.

5. Make sure that at least half of the items in each unit are primary items. Special units may be written for students who need a lot of remedial help. A TJ/EH item may be used to direct a student who is having special problems to a remedial unit.

6. Think about the transportability of each unit you write. Use only the allowable special characters. Make sure none of the lines in a unit is longer than 64 characters. Rigidly adhere to the standard QuATE format and to all rules mentioned in this paper.

7. Make sure the user does not have to spend a lot of time at the terminal solving complex problems or trying to understand difficult text material.

8. Don't write long items if short items will do the job. Long comments are not recommended.

9. Keep answers short so students who cannot type will not spend an excessive amount of time "hunting and pecking."

A DEMONSTRATION UNIT

```
QP:THIS UNIT DEMONSTRATES SIX TYPES OF ITEMS: TJ/EN, TJ/ER,
QC:TN/EJ, TJ/EJ, TH/ER, AND A Q/A ITEM.
QC:
QC:THIS FIRST ITEM IS A Q/A COMMENT ITEM; NO USER RESPONSE
QC:IS EXPECTED.
A@:
QP:WHAT CHARACTER IS USED AS THE CODE TO DENOTE A PRIMARY ITEM?
A@:P
TJ:VERY GOOD!
EJ:NO, THE LETTER P IS USED TO DENOTE A PRIMARY ITEM.
QP:HOW MANY BRANCHING ACTIONS ARE ALLOWED?
A#: 4
TJ: YES, 4 IS CORRECT; J, N, R, AND H.
ER:
QP:HAVE YOU READ PRINTED HANDOUT #1?
A@:YES;Y
TN:
EJ:OK, WE'LL SKIP THE ENRICHMENT ITEM.
QS:WHAT CODES ARE ALLOWED IN THE SECOND POSITION OF A LINE?
QC:NOTE: LIST THE CODES IN THE SAME ORDER AS THEY APPEAR IN
QC:      THE HANDOUT (SEE THE LINE CODE SECTION).
A%:CPS@#%JNRH;9
TJ:YOU PASSED THE TEST.
EJ:THE CORRECT ANSWERS ARE: CPS@#%JNRH
EC:(9 OR MORE RIGHT IS CONSIDERED 'PASSING')
QP:HOW MANY DIFFERENT SYMBOLS COMPRISE THE QUATE CHARACTER SET?
A#:58
TJ:THAT'S RIGHT!
EN:NO.  HERE IS A REMEDIAL ITEM.
QS:THE ALLOWABLE CHARACTERS INCLUDE LETTERS (A-Z), DIGITS (0-9),
QC:BLANKS ( ), AND 21 SPECIAL CHARACTERS.
QC:
QC:NOW, HOW MANY DIFFERENT SYMBOLS COMPRISE THE QUATE CHARACTER
QC:SET?
A#: 58
TJ:GOOD!
ER:ADD 26 + 10 + 1 + 21.
QP:THIS IS THE FINAL EXAMPLE.  IT DEMONSTRATES THE USE OF A
QC:RANGE VALUE IN NUMERIC ANSWER LINES.
QC:
QC:WHAT IS THE APPROXIMATE POPULATION OF THE UNITED STATES?
A#:215,000,000; 15,000,000
TH:THAT'S CLOSE ENOUGH.
ER:THAT'S NOT CLOSE ENOUGH!
LL:END-OF-DEMONSTRATION-UNIT.
```

QuATE EXCHANGE

In order to encourage teachers to become actively involved in the development of transportable QuATE materials, a special-interest group has been organized as part of the Northern California Community College Computer Consortium (NC5). The group conducts workshops, provides technical assistance, and disseminates QuATE materials.

A major effort is underway to implement the original QuATE processing program on a variety of computers of varying sizes. The original version was written in BASIC for a Xerox 560 computer. User responses may include commands which allow the student to "take charge". The commands that have been implemented are $HALT, $JUMP, $NEXT, $REPEAT, and $START. The user may start at any primary item.

QuATE SUMMARY

QuATE - A Standardized Procedure for Describing Structured CAI Materials

The QuATE Procedure:

1. Display a QUESTION (or a comment).	Q - QUESTION
2. Wait for the user to respond.	u - user response
3. Compare user response with stored ANSWER.	A - ANSWER
4. If response matches, THEN-reply and branch;	T - THEN
ELSE-reply and branch.	E - ELSE

Item Types:

1. PRIMARY items are displayed to all users.	P - PRIMARY
2. SECONDARY items are for some users only.	S - SECONDARY

Answer Types:

1. ALPHANUMERIC-character strings (two alternatives)	@ - ALPHANUMERIC
2. NUMERIC-decimal value and range of accuracy	# - NUMERIC
3. PERCENTAGE-character string and criterion score	% - PERCENTAGE

Branching Actions:

1. JUMP to the next primary item.	J - JUMP
2. Continue with the NEXT item.	N - NEXT
3. REPEAT the current item (allow user to try again).	R - REPEAT
4. HALT processing.	H - HALT

<u>The QuATE Format</u>: (Components enclosed in brackets are optional)

```
 1.  QP:character string -OR-
     QS:character string
 2. [QC:character string]
              :
              :
 3. A@:character string [;character string] -OR-
    A#:decimal value [;range of accuracy]   -OR-        [default range: .01%]
    A%:character string [;pass/fail criterion score]    [default score:  70%]

 4.  TJ:character string -OR-
     TN:character string -OR-
     TR:character string -OR-
     TH:character string
 5. [TC:character string]
              :
              :
 6.  EJ:character string -OR-
     EN:character string -OR-
     ER:character string -OR-
     EH:character string
 7. [EC:character string]
              :
              :
```

Notes:

1. Commas and blanks are ignored in numeric and percentage answer lines and responses.
2. Answer lines may not use the semi-colon except as a separator between the first part and the optional second part; with this exception, character strings may contain capital letters [A-Z], decimal digits [0-9], blanks [ ], and 21 common special characters [+-*/,.$'()?=#<>!%&@:;].
3. The maximum allowable line length is 64 characters.

THE MINI-COMPUTER AS AN INTERACTIVE TIME-SHARE INSTRUCTIONAL SYSTEM FOR THE CLASSROOM

David L. Jelden

University of Northern Colorado

Introduction

The proliferation of the mini-computer and the availability of simple program languages make the mini-computer a very viable tool for classroom instruction. Its compactness, ease of operation and relatively low cost warrant its consideration as a possible mode of instruction just as the 16mm film, tape-slide presentations and video tapes which preceded it.

The purpose of this paper is to explain a practical example of a self-contained interactive mini-computer instructional/management system as used in the basic electronics program at the University of Northern Colorado. The system is used as an adjunct instructional tool for the Learner Controlled Education program which is a multi-media, self-paced learner oriented system of learning on an individual basis.

Learner Controlled Education

Learner Controlled Education is defined as a system of instruction that is individually oriented, self-instructional and multi-media in approach. It is based on the premise that students can be taught to analyze the goals of a course, determine procedures or select alternatives that will permit attainment of the goals and carry out those activities suggested.

The learner controlled method is in contrast to the teacher controlled method in which the teacher establishes the goals and determines the approach by which the goals will be reached. In both methods, content is determined by the teacher. The methods differ in terms of who determines or selects the procedures/sources to be used for attaining the goals.

In the truly individualized learning environment, the teacher does not impose his/her method on the student. The teacher's role is to identify goals or course objectives, specify

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© 1978 CCUC

and clarify the alternatives available to the student regarding choices of media and materials available and to annotate the characteristics of those sources so valid student choices can be made. In this setting, the computer becomes an instructional tool and is considered an adjunct to the other materials available in the laboratory resource center. It is in competition to and supplements the traditional textbook or printed materials, 16mm motion pictures, automated tape-slide presentations, autotutorial laboratory tapes, live lecture demonstrations, programmed textbooks and video-tape presentations.

The Computer System

The mini-computer system utilized in this educational setting is an Altair 8800A, 64 K byte main frame with 4 port (terminal) access, 2 floppy disk drives and 3 crt terminals (ADM-1 Lear Seigler) and 1 Decwriter II Printer. The total cost of the system was about $12,000. The program language used is Altair TIME/SHARE extended disk basic. This configuration provides independent access from each terminal to the central processing unit by means of a vector interrupt board and a real-time-clock. The TIME/SHARE basic provides for establishing independent partitions and priorities for each of the peripherals at the time of initializing or powering up the system. The core memory size of each partition can be determined based on need and program length.

Functions Provided by the System

The capabilities of this mini-computer system provide two basic functions. One is management and the other is interactive instruction.

The management function involves one of class enrollment, system clearance and pre/post test scoring and record keeping. It is possible to determine at any time the lessons completed by each individual student and their cumulative scores based on a percentage of their pre/post test performance. This complex record keeping can be done automatically by the computer as the students complete their programs.

It is further anticipated that students can be provided prescriptive guidance in media utilization if programs can be generated to evaluate and process such things as student progress, reading level, numerical and verbal aptitude, personality characteristics, trait-state anxiety levels, etc. against media characteristics such as readability, degrees of illustration, complexity of concept coverage, time of presentation, accessability or availability and degrees of student/media interaction.

It is very possible that some materials might be initial contact in nature where others might be more review or accelerated. Different backgrounds and experiences of students can be used to select the most useful resource for each individual student thus increasing the overall effectiveness of the instructional mode/method available.

Another management function provided for the teacher by the computer is the random generation of tests from a master file. These tests are selected by the computer on request by the teacher.

In the existing data file, 9000 test questions are classified according to course, unit, topic and level. These categories allow the teacher to determine how many questions from a given course over a general unit of study within a specific topic of information to place on the test. Not only that but the difficulty of the test question can be identified and further refined.

At the present time there are three levels of test item difficulty used. The lowest, level 1, is recall or comprehension; level 2 is application or analysis; and level 3 is synthesis or evaluation. Thus it is possible to direct the computer to randomly generate four questions from Course I, Unit 10, Topic B at level 1. The computer then selects these items from all those available under that category and prints it out in hard copy for each individual student. Even if the same request is made for each student the tests will be unique and each student will have their "own" to evaluate their course progress. In addition, the computer will number each test and generate a key on a separate sheet for the instructor if requested.

With this management function, it is indeed possible to truly individualize instruction and evaluation with the computer.

The instruction function of the computer is done through the use of interactive linear/branched programming techniques.

A sample lesson contains the following parts:

1. The management data for test scoring and system clearance.
2. A listing of the objectives for the lesson and any assumptions which are made regarding entry level skills and/or knowledge.
3. A pre-test over the objectives (generally 10-15 questions of the multiple choice variety).
4. An evaluation of the pre-test results and alternatives available determined by pre-test scores.
5. An instructional program with an interactive response mode characteristic.
6. A review of principles, concepts or major points in the lesson.
7. A post-test (same type but different questions than pre-test).
8. An evaluation of post-test results with alternatives and scores posted on computer management file (disk).

System Restrictions and Instructional Guidelines

Based on research, both formal and informal, the generation of 50 instructional lessons and the development of a comprehensive test file, the following conclusions and recommendations are made to those who wish to implement or experiment with a self-contained mini-computer interactive instructional system:

1. You must organize your course content into instructional blocks or teachable units. These should be basic and "universal." That is, they should be able to be understood out of context as a demonstratable or self-contained bit of data (concept--principle). A part of the whole if you please. This allows maximum flexibility in sequencing instruction.
2. The objectives developed from blocks, concepts, principles or teachable units identified should have the extent or level of performance expected clearly stated, the conditions under which evaluation will be made and a clear statement of the specific task. These criteria are basically the same as behavioral or performance objectives which are properly written.
3. Prepare individual lessons or the objectives or content identified. Make each confrontation (lesson) no smaller than 10-15 minutes or no longer than 50-60 minutes for the "average" student. Thirty minutes seems to be the most effective based on fatigue time at

the terminal and concentration levels of freshman-sophomore college students.

4. As a professional teacher, your knowledge of learning theory, instructional materials organization and instructional sequence will apply. Teach those things which are simple, basic and which occur most frequently first and those concepts which are complex and infrequent last. As a general rule, the following guidelines regarding educational theory should apply.

   a. Principles Emphasized Within the S-R or Mechanistic Theory of Learning:

      1) The learner should be active rather than a passive listener or viewer.
      2) Frequency of repetition is still important in acquiring skill and in bringing enough overlearning to guarantee retention.
      3) Reinforcement is important and repetition should be under arrangements in which correct responses are rewarded.

   b. Principles Emphasized Within the Gestalt, Cognitive or Concept/Principle Theory of Learning:

      1) A learning problem should be so structured and presented that the essential relationships are open to the inspection of the learner.
      2) The direction from simple to complex is not arbitrary, meaningless parts to meaningful wholes, but instead from simplified wholes to more complex wholes.
      3) Learning with understanding is more permanent and more transferable than rote learning or learning by formula.

   c. Principles From Personality Theory of Learning:

      1) The learner's abilities are important and provisions have to be made for the slower and the more rapid learners.
      2) The anxiety level appears to be important in determining the beneficial influence of praise or blame.

5. Write the instructional and informational frames for each lesson utilizing the concepts of linear or branched programming. A combination of frames makes a "set" or lesson. The student should not spend more than approximately 30 seconds per frame. Also as new frames are developed, review and reinforcement experiences should be incorporated into each lesson.
6. In writing your instructional frames, each possible response to the item should be anticipated, and even the wrong answers should instruct and lead the student to the next frame. Reinforcement is accomplished in this way. Also, an opportunity should be provided to stop the program at specific key points for re-entry at some future time.
7. Program coding of each written frame in regards to text, response, and answer should be added to give

direction to the computer regarding the next frame or alternative to instruction.

8. Personalize your instruction by providing for the insertion of a first name and by utilizing some humor within the program. Alternative answers, smart remarks, etc., will help relieve the monotony of a strictly verbal program. Use common sense, however, in the number of remarks. Be liberal with praise for a correct answer.*
9. Computer Assisted Instruction, across aptitude levels, is as effective as or better than traditional classroom instruction with respect to student achievement as measured by written and/or performance tests. This also applies to the attrition rate (drop-out level) of the program.*
10. Computer Assisted Instruction, across aptitude levels, has the capability of reducing the learning time of student to a significant degree. Computer Assisted Instruction, when compared to traditional classroom instruction reduced this learning time as much as 30%.*
11. Computer Assisted Instruction students, on the average, like the system or method and the techniques it uses.*
12. The tutorial approach, through Computer Assisted Instruction, is a viable teaching method in the presentation of education materials, concepts and ideas.*
13. The greatest economic saving for Computer Assisted Instruction is in the generation and scoring of objective examinations, thus freeing the teacher for diagnostic work and material preparation in the classroom.*

---

*Information available in a preliminary report from the U.S. Army Signal Center and School, ATSN-EA Computerized Training Systems Project, Dr. Vincent P. Cieri, Educational Adviser, Fort Monmouth, New Jersey 07703. This was on (BASL) Basic Electronics Course used for technical instruction in the U.S. Army Signal School.

REFERENCES

Jackson, William Keith. "A Computer Assisted Instruction System for the Altair 8800 Computer." Published Doctor of Philosophy dissertation, University of Northern Colorado, 1977.

Jelden, David L. "Predicting Success in an Individualized Multi-Media Instruction Program Using Variables of Aptitude and Personality." Staff study, University of Northern Colorado, 1971.

Thompson, Lloyd. "The Feasibility of a Micro-Computer System for Computer Assisted Instruction in Electricity/Electronics." Unpublished Doctor of Education dissertation, University of Northern Colorado, 1977.

A GRAPHIC DESIGN TOOL TO AID IN AUTHORING STUDENT-COMPUTER DIALOGS

Mark Geisert
Alfred Bork
Physics Computer Development Project
University of California
Irvine, California 92717

During the past nine years the Physics Computer Development Project has been engaged in developing a series of student-computer dialogs which attempt to make highly effective use of the computer in teaching situations. A wide variety of pedagogical types is involved, each generated by a good teacher who has ideas about the proper way to assist learning. The material strives to employ the full capabilities of the modern computer.

Focusing on the dialogs rather than on the hardware or software, the Project has developed software and authoring strategies which make it relatively easy for teachers to have ideas implemented. The strategy for preparation we have evolved involves teachers describing the interaction process pedagogically, in a loose flowchart form. Secretaries and student programmers then produce the resulting program. Descriptions of the project[1] and of the underlying software[2] are available.

Preparation of Dialogs

The dialog software which has evolved under the demands made by the teachers is primarily macro-based, with new macros ("procedures" in the assembly language employed) generated on the basis of pedagogical demand. The macros are concerned with display, with analysis, and with other aspects of preparing computer-based learning material. FORTRAN subroutines are also utilized in parts of the programs, particularly where extensive calculations are needed. The programs are long, making use of the overlay facilities of the computer, a Sigma 7.

Almost all of our dialogs involve extensive use of graphics, with the Tektronix 4013 as the display device. Undergraduates at Irvine do most of the programming, working either from files entered by secretaries or from the author-generated flowcharts. The

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

effective use of the authoring strategies is indicated by the wide body of highly interactive, computer-aided learning material we have produced.

The Problem of Graphic Design

One of the most tedious aspects of programming dialogs involves certain types of visual material to be displayed on the screen, particularly in combinations where intimate blends of graphic and alphanumeric information are present on the screen at the same time. The visual appearance of the dialogs is, we believe, an important factor in their success. The screen is a new medium, not a printed page. We can not only affect motivational issues by the screen layout, but we can also aid reading comprehension.

The graphic and alphanumeric components must be closely matched in the displays, with no overlapping. To program these complex graphic pictures with conventional means, by referencing the screen with coordinates, is not simple even though we have developed a variety of software to ease the task (including macros that let the programmer refer to the screen by means of character positions). Experience with many recent dialogs indicates that programming of this type, graphic composition, is one of the major activities of the programmers. For effective material we need skilled programmers willing to take the patience to achieve a careful blending of pictorial and text material.

The Graphic Design Capability

In connection with developing a student-computer dialog, a student working on that dialog produced a set of APL functions to assist the dialog writer in creating complex blends of pictorial and alphanumeric material. The capabilities are interactive. We stress that the final product for our dialogs is not an APL program but a set of macro calls.

Our experience shows that APL can be an extremely effective aid in assisting with dialog construction. This paper describes the existing facility, one that has evolved and been used in many dialogs over several years. The "feeling" of these APL functions is similar to the graphic language, 'RAP', developed at the California Institute of Technology. We call the workspace involved DPL.

The APL we employed (available on the Sigma 7) has full graphic capability built into it; that is, the graphic software is not a set of separate APL functions in a workspace. Our project was responsible for adding the graphic capability to APL, primarily for use by students in connection with problem solving activities, and the complete system is now distributed and maintained by Xerox (now Honeywell) as part of their program product APL.[3] We have argued elsewhere[4] that APL is a particularly useful language for use in graphics, because of its powerful array handling and constructing capability. If graphic capability is embedded in APL, it is reasonably efficient. The full power of APL is available for the system described here. The workspace has been transported to several other systems, including one IBM system.

Overview of System - An Example

Perhaps it is easier to show how the APL authoring aids work if we go through an example. Let us suppose that the task at hand is to display the following equations:

210

$$X_1' = AX_1 + BX_2$$

$$X_2' = CX_1 + DX_2$$

inside a geometrical figure that emphasizes the equations. These equations, as with many equations in physics and mathematics, have subscripts.

The programmer enters APL and loads the APL workspace, DPL, which contains the necessary facilities. The programmer might then enter the two equations giving each one a name. He or she types for the first,

```
EQ1 ← XTEXT
```

The first result will be that the graphic cursor will be turned on. The author or programmer then points to the beginning location for the first equation, indicating the lower left edge of the first letter. The computer queries as to what text is to be written at that location. The symbols "/" and "\" allow the existence of superscripts and subscripts. Note that this text, the equation, has a name associated with it, EQ1, and can be placed in various locations by referencing that name. We can show its current "value" by typing DI EQ1.

In a similar manner a second equation, to be called EQ2, is entered. Now we wish to place some geometrical object around these equations to emphasize their importance in the material. We do this as follows:

```
FIGR ← XPOLY
```

XPOLY is another built-in function. In general, the X in front of the function name indicates that part of the input is to come from the graphic cursor. Functions without the X accept numerical arguments referring to locations on the screen. XPOLY will query the student as to the size of the polygon (largest screen dimension is 1024 units) and the number of sides. Note that the polygon has a name too, and can be referenced by that name. The author might decide that he doesn't like the current location, so he might move the figure as follows:

```
FIGR2 ← FIGR AT    [0]
```

Then the crosshairs will again be turned on, and we can point to a new center for the figure; $\boxed{0}$ is the graphic input operator in the Xerox APL. We now have both figures available, the old one and the new one, and each is identified by a separate name. We might have simply replaced the first figure by the second by using the name FIGR over again.

After these subpieces have been moved until we are happy with the visual appearance, we might want to construct a <u>composite</u> object containing both the equations and the graphic material associated with the equation. We can do this with no difficulty, by defining a new "object," as follows:

```
TOTAL ← FIGR2 AND EQ1 AND EQ2
```

This object, called TOTAL, is a combination of graphic and alphanumeric information, drawn in the order indicated. It can itself be moved around. Let's suppose that the author decides that it really appears a little bit too high on the screen. He or she proceeds:

NEW ← TOTAL AT $\boxed{0}$

The crosshairs are again turned on, and again the user points to the center of the desired location. A new object has been defined in this process, and perhaps this is the one desired. If further movement or modification of either piece or the total object is necessary, that is still possible; they are all named entities.

We could proceed in this manner to construct a complex picture involving both pictorial and alphanumeric details. The facility is to some extent like that ordinarily available to a graphic designer.

Up to this point, no attempt has been made to generate code. Each named object is an APL vector with the information for creating the graphic curve or alphanumeric information. The APL function to generate code is called SAVE.

The net result of this process is to create a file called GLOP. This file has an initial label, EQUATION, and uses for internal labels the root, POINTS.

Here is what the file created by DPL actually looks like:

```
EQUATION MARK
         WINDOW     (FS'0',FS'0'),(FS'8.25',FS'6.29')
         SCALE      (FS'0',FS'1024'),(FS'0',FS'780.15')
         CURVE      (POINTS1X,POINTS1Y,8)
         SSETPOINT  (FS'374',FS'344')
         OUT        '1     1     2'
         SSETPOINT  (FS'360',FS'351')
         OUT        'X     = AX  + BX'
         SSETPOINT  (FS'388',FS'358')
         OUT        ''''
         SSETPOINT  (FS'374',FS'264')
         OUT        '2     1     2'
         SSETPOINT  (FS'360',FS'271')
         OUT        'X     = CX  + DX'
         SSETPOINT  (FS'388',FS'278')
         OUT        ''''
         USECT      *PURE
POINTS1X DATA       FS'691',FS'748',FS'586',FS'326',FS'164',FS'221'
         DATA       FS'456',FS'691'
POINTS1Y DATA       FS'416',FS'280',FS'172',FS'172',FS'280',FS'416'
         DATA       FS'476',FS'416'
         USECT      *ALFRED
```

The details will be understood by someone familiar with the macro-based software for creating dialogs developed in the Physics Computer Development Project. The commands are macro names. WINDOW and SCALE set the screen coordinates and determine the area in which plotting is to take place. The SSETPOINTS move the CRT beam around, and the OUTs do the actual writing of alphanumeric material. CURVE outputs graphic material, using the values that DPL has constructed and placed in the DATA statements.

This, of course, is not a complete program. Many such segments, and code produced by more conventional means, will be merged using the editing facilities on the machine to produce a complete program.

Capabilities

A sequence to be used many times in the program can be created as a subroutine, and its position on the screen can even be determined at runtime based on what is already on the screen.

The graphic objects that can be created by DPL functions include lines ('regular', 'thick', or 'dashed'), points, arrows, rectangles, polygons, ellipses, and curves defined in the form F(X) vs. X or parametrically as Y(T) vs. X(T).

Text information may be created as 'regular', screen-centered, right- or left-hand justified, and may contain arbitrarily complex subscripting and superscripting.

DPL functions which change the content of graphic objects allow moving the object up, down, left, or right, or any combination of these, relative to the object's current position; moving the object to a specific location; expanding the object relative to its center or some arbitrary point; rotating the object around its center or some arbitrary point; reflecting the picture through some line; and obtaining a subcurve of a given curve.

A function is provided for creating a picture on the Tektronix graphic tablet. A collection of pictures may be saved and restored from disk independent of the APL workspace management. An editor is provided. A 'DO' statement can be used to create pictures needing iterative components.

Many other capabilities are available in DPL. The "DPL User's Guide" describes them in detail along with many examples ready to be tried at the terminal.[5]

DPL thus provides an extensive library of graphic functions for the quick and interactive construction of student-computer dialogs.

References

1. Bork, A., The Physics Computer Development Project, EDUCOM Vol. 10, No. 4/Winter 1975.

2. Bork, A., Teaching Conversations with the Sigma 7.

3. Xerox APL Language Reference Manual.

4. Bork, A., APL as a Language for Interactive Computer Graphics, Proceedings, APL6.

5. Geisert, M., DPL User's Guide.

THE COMPUTER, THE INDIVIDUAL AND SOCIETY: A COURSE FOR SECONDARY TEACHERS

Stanley F. Robinson

Eastern Washington University

Introduction

Computer technology touches our everyday lives. The near future holds the promise of computers as a household appliance. The growing interaction between computers and our lives presents society with important public policy issues. At Eastern Washington University we have developed an upper division course to provide information regarding applications of computers and the decisions society is likely to face regarding computers. We feel that the course is particularly appropriate for prospective and inservice teachers who should have the background to discuss these issues with their students. However, the course is open to all students who meet the prerequisite of at least one previous course involving computers or computer programming. This minimal prerequisite results in students prepared to appreciate the impact of the computer and allows us to discuss the issues at a more sophisticated level than in a strictly introductory course.

In comparison with computer and society courses reported previously, the course at EWU would most closely resemble courses provided for computer professionals (5) or those presented in the seminar form. (2, 6)

Objectives

Three objectives are given for the course.

1. An understanding of the use and effects of the computer in our society.
2. Understanding the capabilities and limitations of a computer.
3. Recognition of the need for public concern and user responsibility in current and future applications of the computer.

Course Content

The course has three basic divisions in the material that is studied.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

1. Historical background
2. Current applications of computers
3. Issues for society

Historical Background

This segment of the course is relatively short. The intent is to provide a historical perspective to current activities in computer use and future directions. The emphasis in this segment is on the development of information demands in modern society and the development of computers to meet this demand. (1, 3, 8)

Current Applications of Computers

Textual material and outside readings are used extensively during this segment of the course. A number of simple computer programs are used to demonstrate computer applications.

Since the students involved in this course have had some previous exposure to computers in their previous course work, time is not spent in elementary programming or in the nature of computer hardware and software. Some time is spent developing file handling commands since many of the issue areas involve the capacity of the computer to handle and manipulate data files.

The areas where computer programs are used to illustrate applications are as follows:

1. Programs to illustrate the computer used for computations
   a. PRIMES - This program will compute and print a list of the first N primes.
   b. TRIANGLE - This program will solve an oblique triangle.
   c. GCDLCM - This program will find the GCD and LCM of two given numbers.
2. A program to illustrate the power of the computer to retrieve and handle data.
   a. EARTH - A developmental program from the Northwest Regional Educational Laboratory, 710 S.W. Second Ave., Portland, Oregon 97204

      This program is organized around a data base of information related to global energy use. Twenty-four items of information are stored for each of the 83 largest countries in the world. The program allows the user to obtain tables of up to 5 variables at a time and to obtain rank ordering of countries by a variable.
   b. Arrangements are made with our library to have the class observe a bibliographic search in the Lockheed DIALOG system.
3. Operation of a simulation
   a. ENERGY - Another development program from the Northwest Regional Educational Laboratory. This program asks students to balance energy supply with a number of demand sectors and at the same time maintain general satisfaction, economic well-being and pollution levels.

4. Conversing with the computer

   a. An ELIZA-type program provides an excellent exercise to demystify the computer by having the student study this program after they have operated it.

5. Reservation system

   a. AIR - A program demonstrating use of the computer to make airline reservations. The program sets up a file with passenger name, date and flight number and provides an opening for a discussion of the type of information on individuals placed in computer files as a result of some computer applications.

6. Electronic transfer of funds

   a. BANK 1 and BANK2 - Two programs to illustrate the ease of transfer of money by computer terminal and at the same time the need for identifiers for security.

7. Use of the computer for drill

   a. BINARY - A drill that asks the student to convert five binary numbers to base 10 followed by five base 10 numbers to change to binary.

   b. FACTOR - A drill in factoring trinomials, $AX^2 + BX + C$.

   c. LINEAREQ - A drill in the solutions of equations of the form $AX + B = C$.

Issues for Society

The major portion of the course is spent in the study and discussion of ways that computers may affect our social fabric. As part of this segment of the course each student is expected to study and write a term paper on two or three of the questions presented to the class. An extensive bibliography of current literature is made available to the student.*

Current issues and news items often determine the direction of the course. In the summer of 1977 a great deal of time was spent on the issues raised by the newly released report of the Privacy Protection Study Commission.

The issues posed are subject to debate and are posed to increase the students' awareness of the value judgements required of society. The following is a partial listing of the questions presented to the students for consideration.

1. List some jobs that could easily be eliminated by automation. List some that would be difficult to eliminate. Discuss the effect the elimination of these jobs will have on society and the individuals involved in them.

2. List some new employment opportunities you see flowing from the increased use of computers. What impact do you see on individuals and on education as a result of these new occupations?

3. List some occupations you see significantly changed by the use of computers. What impact do you see on those employed in these occupations?

---

*Available from the author at the Department of Mathematics and Computer Science, Eastern Washington University, Cheney, Washington 99004

4. What controls should an individual have over records of his habits of purchasing and expenditures gathered by his use of banking and credit card facilities? What agencies, if any, should have access to these records and on what basis?

5. What controls should be available to the individual as to the accuracy of information on computerized records?

6. What records or receipts should the consumer receive from electronic transactions?

7. Current checking practices allow individuals to write checks and deposit funds a day or two later and to stop checks for defective goods or unfulfilled services. If electronic funds transfer eliminates any time lag, what will be the effect on individuals? Do you believe an electronic funds transfer system should have a built-in time delay?

8. How should individuals using automatic funds transfer systems be protected from error?

9. If financial (or other) institutions are interconnected, what controls should be placed on the flow of individual records from one institution to another?

10. In the United States computerized files containing personal information have been created by a number of agencies. The Defense Department, National Criminal Information Center, National Driver Register Services and the Social Security Administration for a few examples. Discuss the positive benefits that result from governmental use of these files. Also discuss any possible injury that might occur to individuals if free access were given to these files. What controls of access ought to be maintained on these files?

11. Many of the records maintained by the government are needed for research and statistical studies. How do such activities complicate the privacy of individuals? What protection should be present to protect individual privacy? It has been suggested that these activities would be less expensive and more effective if government files were on one computer system. Discuss the benefits and difficulties you see with this suggestion.

12. In the past decade a number of agencies have maintained computerized files on individuals considered a security risk. What agencies should maintain such files? What criteria should determine if an individual should be included? Should an individual be advised if information is collected about him? Should he be allowed to rebut information in such a file?

13. To what extent do you think school records should be available to police agencies to locate potential criminals, drug users or others who might create problems to society?

14. What benefits and what dangers do you see if records for each individual are maintained by a universal identifier such as social security numbers?

15. In what ways could individuals make use of a home computer?

16. Why might an individual wish to keep data files on others such as his neighbors? Should all computers that keep data files on individuals be registered with some government agency?

17. To what extent should police or other government agencies have access to data files maintained by a business or an individual?

18. How close are we to computerized war? Do automated weapons systems make killing more impersonal? Do you think they make war more or less likely?

19. Do you think computerized war games and the micro-processor TV games on the market make war more or less acceptable?

20. Considering the positive things computers can do and the history of human contact in education, discuss what you feel should be the proper role of computers in education.

21. Considering the positive things computers can do and the history of human contact in medicine, discuss what you feel should be the proper role of computers in medicine.

References

1. Arbib, Michael, Computers and the Cybernetic Society, New York: Academic Press, Inc., 1977.

2. Decker, Craig, "Planning and Experimental Seminar on 'Computers, Society, and the Future'", CCUC/2 (Computers in the Undergraduate Curriculum), Dartmouth College, Hanover, New Hamshire, June 1971.

3. Gotlieb, C. C. and Borodin, A., Social Issues in Computing, New York: Academic Press, Inc., 1973.

4. Hertlein, Grace C., "Computers, Society, and Man", Proceedings, CCUC/5, Washington State University, Pullman, Washington, June, 1974.

5. Hertlein, Grace C., "Engineers, Computers, Social Responsibility", Proceedings of the Computer Science and Engineering Curriculum Workshop, IEEE, 1977.

6. Langland, Lois and Weingarten, F. W., "Social Implications of Computers at Scripps College", Proceedings CCUC/2, Dartmouth College, Hanover, New Hampshire, June, 1971.

7. Mowshowitz, Abbe, The Conquest of Will: Information Processing in Human Affairs, Menlo Park, California: Addison-Wesley, 1976.

8. Van Tassel, Dennis L., The Compleat Computer, Palo Alto, California: Science Research Associates, 1976.

9. Weizenbaum, Joseph, Computer Power and Human Reason, San Francisco, California: W. H. Freeman and Co., 1976.

10. "Personal Privacy in an Information Society", Report of the Privacy Protection Study Commission.

11. "EFT in the United States", The Final Report of the National Commission on Electronic Fund Transfers.

UNDERGRADUATE PROGRAMS TO INCREASE INSTRUCTIONAL COMPUTING IN SCHOOLS

J. Richard Dennis

Department of Secondary Education

University of Illinois

Introduction

This paper presents a position regarding the delivery of technological instructional methods to teachers and schools. The specific technology addressed here is instructional applications of computers (IAC). But the discussion can be viewed as an instance of a much more general problem in teacher education. That of promoting the development of alternative professional skill specialties among the teaching corps.

During the recent history of education several technologies have been developed which early proponents viewed as a means of changing significantly the location and/or nature of the delivery of education to the masses. Among these technologies have been instructional films, educational television in either broadcast or closed circuit modes, and programmed instruction. Behind the classroom scene, thermal printing and other new reproduction technologies have offered significant advancements in preparation of classroom materials. A close analysis of these technologies in schooling today reveals a wide variance in the incidence and impact that has accrued in each instance. Modern materials reproduction technologies have found favor and regular use in nearly all schools, large or small, and in seeming disregard of levels of financial resources. Educational films widely are available in a variety of subjects but many teachers and schools make scant use of them particularly if budgets are tight. Programmed instruction, following a flury of activity in public schooling in the fifties and sixties, scarcely is mentioned or observed in school settings today. And educational television has not lived up to its advanced billing as a means for major improvements in classroom instruction.

Reasons for differences in implementation among advanced materials preparation technologies and the other "teaching technologies" are not difficult to articulate. Foremost is the fact that modern reproduction technologies merely were offering improvements in the operation of the prevailing classroom and

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

teaching role. The other technologies, for optimal impact, call for changes in classroom structure and management, and more important, changes in the actions and roles of teachers. In each of these cases the implied changes in the teacher's role and day-to-day activity are of a magnitude that long duration, very intensive training experiences offer the only promising avenue of permanent implementation. Prevailing professional habits and methods are notoriously difficult to overcome and to replace.

Teacher education takes place at two levels, preservice and inservice. At the preservice level the potential teaching corps would be able to devote almost exclusive effort to training and education. This professional development can be organized to be very intensive for a long duration, partly because the participants have little else on which to devote their attentions. Contrast this with continuing professional education at the inservice career stage. Here teacher training takes place either through school organized, short workshops, through semester long extramural courses, or through attendance at brief summer sessions. In neither of these modes does one find the combination of the opportunity to devote almost exclusive attention and the opportunity to do so for an extended period of time. Long term, intensive professional development could take place at the inservice level if sabbatical practices were more common. But, present funding pictures virtually preclude any such opportunities in the foreseeable future.

The past two decades have seen a new technology develop which, again, proponents have hailed as a means of revolutionizing the location and delivery of schooling. Computer-based instruction, or more descriptively, teaching with computers, has seen two or three fluries of promotion as a future style of instruction. Again, however, we are witnessing a technology which, for optimal benefits, appears to dictate major changes in the role of the teacher, and in the structure and management of the classroom.

Both in the cases of educational television, instructional films, and programmed instruction, and more recently of computer-based education, efforts at dissemination to schools largely have ignored preservice professional training for more than a casual introduction to the topics. Few, if any, efforts have been undertaken to develop, at the preservice level, professional knowledge and competence in these alternate instructional styles. And yet, as mentioned before, the preservice career stage essentially is the only opportunity in the total career picture for focusing attentions intensively for an extended duration.

A major factor accounting for a lack of attention to alternate professional skills may be the sometimes tacit, but powerful, pressures toward uniformity in professional training. For all practical purposes, the only diversity permitted in teacher education programs is in the subject taught, but even here divergent specialties within the field are not greatly encouraged. To reverse this picture without the need of major changes in teacher certification practices, we are arguing for a new view of the undergraduate minor area of concentration. Historically, teacher educators have argued for minor concentrations in alternate subject fields at the preservice level. State teacher certification boards recognize such concentrations and employing school administrators frequently make employee selections on the basis of the potential for alternate assignment of a prospective employee. We do not disagree with this practice as one alternative, but are arguing that as the only alternative this practice eliminates opportunities for structured, supervised development of prospective personnel in a variety of equally important professional techniques and skills. Only at the preservice level can training efforts requiring sustained, intensive concentration of both the trainer and the trainee be realized. Further, we are arguing that by developing an array of alternative professional concentrations at the undergraduate level, and encouraging diversity (but demonstrated expertise) among those entering the profession (even though they may be few in number), the long term impact on schools and the educational processes promises to be much greater than that deriving from the totality of casual training efforts at the inservice level.

The Case for Instructional Computing

The specific preservice professional skill specialty we wish to discuss here is that of applying computers to instruction. That this is a potentially productive area for training is witnessed by several recent studies. Bukoski and Korotkin (1976), in a survey by the American Institute for Research found approximately 28% of the secondary schools in the nation somehow active in instructional use of computers. This finding is supported on more regional levels as well. For example, Dennis, Dillhunt, and Muiznieks (1977) found 35% of the secondary schools in Illinois reporting instructional computing. Both studies found growth rates for such activity (in terms of number of new schools reporting activity) averaging around 4% per year over the past 5-10 years.

Two major characteristics of instructional use of computers are reported by both of these studies that support the position presented here. First, in schools reporting instructional computing, the majority of the activity centered around teaching computer programming. Second, again a majority of the active schools reported very low levels of involvement across their total teaching core. This suggests that perhaps instructional computing resources are underutilized in many schools, and that the portion of the student body or number of courses receiving benefits from such technology is rather small. Even though costs for computing resources appear to be declining such costs for a typical secondary school are add-on costs. No teachers are (or should be) replaced by introduction of computers into the instructional process. But, when budgets become tight, as is the case in many schools of late, those add-on costs most easily justified, and most likely to survive scrutiny, are those clearly benefitting the larger portions of the school community.

From still another point of view, using computers in the instructional process, as those teachers actually doing same will atest, is an activity that requires concentrated energy in order to become proficient. Only a few, very eager teachers have been able to muster such concentrations of energy and/or time on the job (inservice). We are arguing for an alternate route to impacting on schools in this area through functional expertise derived by a diverse segment of the potential teaching corps at the preservice career stage.

A Professional Minor in Instructional Applications of Computers

The University of Illinois, Urbana, has instituted a program which addresses the growing role for teaching personnel with credentials in a conventional subject area and functional competence in using computers in the teaching of that subject. The general characteristics of this program are as follows:

1. An opportunity to acquire the technical background in computer programming adequate to the demands of instructional computing.
2. An intensive "whole school" view of instructional computing, coupled with a discipline specific instructional computing methods experience.
3. Hardware and software background adequate for conditions likely to be found in schools today, but also adequate for dealing with the changes likely to be experienced over a period of 5-10 years.
4. A concentration on the relevant findings from research areas such as learning, teaching, curriculum development, and evaluation. Too often, research on the teaching/learning processes is discounted as having virtually no value to practitioners. The position of this program is that some research in education is pertinent to effective design/delivery of computer-based instruction.
5. An opportunity to demonstrate proficiency in instructional computing through a task-centered practicum. This would be comparable to a micro-teaching experience so often recommended as a precursor to practice-teaching in a discipline, but would concentrate on the tasks of teaching with computers.

6. A controlled practice teaching experience in a setting where computers are available for use in the instruction of the candidate's teaching major.

From a more detailed point of view, the program consists of six parts, five of them required of all participants, extending for at least four semesters. Following is a description of these parts.

I. Background in Computer Science

Normally, this consists of a beginning and an advanced undergraduate programming course. Generally, the goal of this part of the curriculum is to develop reasonable skill as a computer programmer. In addition to these two courses, there is a three-hour elective in Computer Science selected from among advanced systems programming, hardware, numerical analysis, logic design, or an applications course relevant both to computing and to the teaching major.

II. Foundations of Instructional Computing (a one semester course)

A. The literature and resources of instructional computing with emphasis on integration of this technology with prevailing instructional practices.
B. Survey and analysis of the variety of educational uses of computers and general characteristics of each, including computer managed instruction, drill and practice, tutorial, simulations, instructional games, problem solving, evaluation and test item banking, record keeping and processing.
C. Studies of various hardware characteristics, capabilities, and the effects on school application. Consideration of specific instructional activities as they would be implemented by alternate hardware systems, and the accompanying changes in teacher behavior and responsibilities. The systems considered range from programmable calculators, through micro-computers, to large scale network systems. Included is a thorough consideration of both batch and interactive instructional applications opportunities and techniques.
D. Reviews of relevant research on learning, teaching, curriculum, and evaluation. Detailed studies of the implications of such research for optimal design of instructional computing.
E. Social and legal implications of computers in schools and society; privacy; computer security; computers in the home and community.
F. Models of system management and operation in schools; implications for personnel access and management.
G. An introduction to issues in hardware/software procurement and maintenance.
H. Characteristics and techniques of materials development for IAC--experiences with a variety of prevailing authoring language styles (e.g., TUTOR, BASIC, CAISP, PILOT, IDF, etc.).

.structional Computing--The Content Specific View

(An array of one semester courses organized around the teaching majors of the students.)

1. Critical survey and analysis of instructional computing within the content area of the teaching major.
2. Integrating responsive systems with classroom instruction; role analysis and designation within classroom management models peculiar to the discipline.
3. Design and evaluation of courseware in the teaching major.
4. Automating evaluation and record keeping in the discipline.
5. Projects and materials design.

IV. Task-Centered Practicum in Instructional Computing

An opportunity for repetition of fundamental practice skills. Teaching competencies are defined in the following categories:

A. Preparation and use of controlled, interactive, computer-based instruction.
B. Preparation and use of computer controlled, interactive, computer-based instruction
C. Design, implementation, and utilization of simulations.
D. Facilitating computer problem solving.
E. Designing student evaluation and record keeping.
F. Resource allocation and management.

V. Practice Teaching

Students practice teach in schools having instructional computers available. Of course, participants all practice teach in their subject major.

VI. Thesis Option (elective)

Students desiring a unique experience may elect to compare an instructional computing thesis. Each such project includes the following components:

A. Identification and definition of an instructional application of computers.
B. Studies of relevant literature.
C. Project development, testing, and formative evaluation.
D. Project implementation and summative evaluation.

The goals of this program are derived from the experiences of practitioners. An intensive survey was conducted, following the Delphi procedure, among an extensive list of persons identified as being knowledgeable of, and/or having had extensive experience with, instructional computing. The purpose of the survey was to articulate and to achieve a degree of unanimity on the important knowledges and skills characterizing proficient instructional computing. From this study, the following were rated by a majority of those surveyed as somewhat to very important to expertise in instructional computing:

1. Familiarity with computerized teaching materials (i.e., instructional programs) in a variety of fields.
2. Ability to integrate computerized teaching materials into a course.
3. General knowledge of the functioning of CMI (computer-managed-instruction) systems.
4. Understanding of effective design of drill and practice materials.
5. Ability to apply computerized drill and practice in a variety of teaching situations.
6. Familiarity with computer simulations and models.
7. Experience in preliminary design and construction of a simulation.
8. Knowledge of the uses of simulations as teaching tools.
9. Ability to evaluate the effectiveness of a course that uses computerized teaching materials.
10. Ability to determine the computer needs of a school.
11. Ability to draft specifications (request for proposals) which set down the needs and desires of the school and invite proposals/bids from potential suppliers.
12. Ability to be highly critical of suppliers proposals and their equipment.
13. Ability to assemble data about proposed equipment to facilitate decision-making (costs, performance data, hardware characteristics, software support, etc.).
14. Familiarity with instructional games.
15. Knowledge of how to use instructional games appropriately and effectively in teaching.

16. Physical familiarity with computer equipment, i.e., everyday operation and use of a range of different machinery.
17. Knowledge of trouble-shooting procedures and means of access to professional help, i.e., knowing how to determine if a piece of equipment is ailing, and if it is, knowing whom to call to fix it.
18. Knowledge of sources for computer materials.
19. Competence to improve less than adequate instructional computer programs.
20. Ability to evaluate the effectiveness of instructional computer programs.
21. Ability to instruct others in the social role and impact of computers in society.
22. Knowledge of alternative uses of computers in schools, e.g., as class-record-keepers, term-paper-editors, etc.
23. Awareness of the value of involving students in the development of computerized instructional materials.
24. Knowledge of processes of involving students in instructional materials development.

Implications for the Undergraduate Institution

For most teacher education institutions, enrollment projections over the next several years clearly suggest major reductions in program size. In this type of environment, an institution could not undertake to operate specialty programs in large numbers. It would be an unusual campus, indeed, where teacher education enrollments supported concentration programs in all of the conventional teaching majors and minors and in the areas of instructional films, educational television, programmed instruction, instructional computing, etc. What is implied in a proposal such as presented here, however, is an opportunity for institutions to develop individual program specialties and distinction. In doing so, a particular institution has an opportunity to establish, for itself and its teacher education graduates, a unique reputation.

Along with declining teacher education enrollments has come a situation either of excess teacher education staff or of grossly underutilized staff. In such an environment, institutions can consider selective efforts to redirect staff and at the same time to achieve significant improvements or additions to existing programs. A decision to implement an Instructional Computing specialty program such as described here presents a unique opportunity for staff development. The Computer Science component is not seen as a major problem since a reasonable offering in computer programming is quite common among post-secondary institutions today. What is needed is the staffing for the "foundations" experiences in instructional computing. Here is a role for a professional development sabbatical of roughly a year's duration to gain the necessary background in technology, alternative instruction, etc. A person noramlly working in the areas of teaching methods and/or curriculum in the fields of mathematics, science, or business education particularly is suited to the role of resident instructional computing "expert." A year's intensive involvement in instructional computing at one of several centers of such activity will provide the necessary expertise for initiating, coordinating, and auxiliary staff development for such a program. But, many institutions would need less deliberate effort because one or more staff members already casually may be into instructional computing.

The general strategy just described would be appropriate for implementing an undergraduate teacher specialty in any of the alternative instructional methods mentioned earlier. The purpose of this paper, however, is to encourage some more such efforts in the specialty of teaching with computers. Many resources for such activity are available in a surprising number of schools today. And, generally, these resources are not utilized to their full potential. The future of computers in schools is dependent upon the development of a larger cadre of more highly skilled practitioners.

Bibliography

1. Bukoski, William and Arthur Korotkin, "Computing Activities in Secondary Education," in Educational Technology, Vol. XVI, No. 1, January 1976.
2. Dennis, J. Richard, Clifford Dillhunt, and Viktors Muiznieks, "Computer Activities in Secondary Schools of Illinois," Illinois Series on Educational Applications of Computers, No. 24, University of Illinois, Department of Secondary Education, 1977.

WHAT MAKES A COMPUTER-LITERATE COLLEGE?

Beverly Hunter

Human Resources Research Organization

ABSTRACT

Twenty-four colleges and universities were selected to illustrate a variety of goals and programs aimed at increasing the level of computer literacy among students and faculty. In this paper, components of these exemplary programs are summarized. Roles of different organizations such as computer centers, computer science departments, and educational research and development groups are described. Computer literacy courses are reviewed. Indicators of the impact of these programs are provided.

INTRODUCTION

"Computer literacy" is a very general concept that may be defined as: "Whatever a person needs to know and do with computers in order to function competently in our society." The kinds of knowledge, skills, and attitudes a person needs depends upon his/her vocation, avocations, and lifestyle. One or more of these kinds of skills and knowledge may be required by a computer literate person:

(1) skill in writing algorithms and computer programs;
(2) knowledge of computer applications in one's field of endeavor;
(3) understanding of computer systems and their impact on society.

Another important dimension of computer literacy is a person's attitude of control over computers and computer systems.

EXEMPLAR INSTITUTIONS

Some colleges and universities of every size and type have formally recognized computer literacy as an academic goal, and have established programs to meet the literacy needs of students and faculty in all fields of study. Twenty-four such colleges and universities were selected to illustrate a variety of computer literacy goals and programs. These institutions, among others, participated in a

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

study, "Exemplary Institutions in Academic Computing."[1] The institutions range from very small private colleges to large state universities. A list of the Exemplars is shown as Figure 1. They were selected as Exemplars because they (a) have demonstrated institutional commitment to academic computing, (b) have a relatively high level of computer literacy throughout the campus for their type of institution, and (c) have established programs specifically addressed to increasing the computer literacy among students, faculty, staff, administration, or the general community.

CATEGORY 4: COMPUTER LITERACY PROGRAMS FOR STUDENTS, FACULTY OR COMMUNITY

Colleges and Universities With Student Enrollment Over 6,000 FTE

Auburn University (AL)
University of California, San Diego (CA)
Mankato State University (MN)
Rutgers University (NJ)
University of Illinois, Urbana (IL)
New York Institute of Technology (NY)
University of Texas, Austin (TX)
University of Texas, El Paso (TX)
University of Wisconsin, LaCrosse (WI)

Colleges and Universities with Student Enrollment Under 6,000 FTE

Colorado School of Mines (CO)
Fairfield University (CT)
Grinnell College (IO)
U.S. Naval Academy (MD)
Carleton College (MN)
Northern Montana College (MT)
Dartmouth College (NH)
Bennett College (NC)
Denison University (OH)
Cameron University (OK)
Bucknell University (PA)
Moravian College (PA)
University of Tennessee, Chattanooga (TN)
Trinity University (TX)
Evergreen State College (WA)

Figure 1. Exemplar Colleges and Universities

ORGANIZATIONAL ROLES

Computer literacy, being a relatively new requirement for education, did not have a "home" in traditional academic disciplines and departments. The leadership and responsibility for literacy has emerged from several different organizational entities.

Administrative Leadership

Several exemplar institutions cited the leadership of the administration as a key element in academic computing. Dartmouth is a well-known example, with the leadership provided by President Kemeny in the 1960's. Other smaller institutions have made a formal commitment to literacy. For example, the Evergreen State College Bulletin states:

---

[1] This study was partially supported by the National Science Foundation, Grant No. SED 76-15399, to the Human Resources Research Organization. Dr. Robert J. Seidel is Principal Investigator of the Project, Beverly Hunter, Co-Principal Investigator. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the author and do not necessarily reflect the view of the National Science Foundation.

> It is important for every educated individual in today's world to know something about computers and the way they are used to process information and make decisions. Computers directly influence our lives in an increasing variety of ways. The crucial issues involved in society's use of computers are far too important to be left only in the hands of experts.

Computer Center

Computer center directors and staff have played major roles in establishing and conducting computer literacy programs. They have not only tried to provide faculty and students with access to appropriate computing facilities, but also have provided training that faculty need to become computer literate. Richard Storer, Director of the Center for Computer and Information Services at Rutgers University, dates the inception of a formal computer literacy program at 1971, when the Center began its education series for faculty. The CCIS Education Series is attended by faculty from all departments. The programs contain seminars on basic introductory computer concepts such as "Everything You Always Wanted to Know About Computers But Didn't Know Enough to Ask," "Introduction to Statistics," and "Introduction to Programming." Similarly, Mankato University Computer Services Department began offering short courses in computing to faculty and staff in 1962, and the University of Texas at Austin computer center offered short courses in 1965. The U.S. Naval Academy in 1966 provided a two-month summer program for faculty on instructional computing applications.

The Computer Center at the University of California, San Diego, established a formal Computer Literacy program in 1975-76. The objective of this program is to provide an opportunity for any registered student to become familiar with computers and computing. The program is aimed at the social science, fine arts and humanities majors, on the assumption that mathematics and physical science majors will be introduced to computers in the normal course of their instruction.

Academic Computing Services Department

Establishment of an academic computing services department is considered by some to be a prerequisite to campus-wide literacy. A recent example is the University of Tennessee at Chattanooga, which established such a department in 1975. Prior to establishment of this department, students and faculty had no advocates among the computer center and could not compete with administrative computing demands.

Computer Science Department

About half of the exemplar institutions award undergraduate or graduate degrees in Computer Science. Many of these computer science departments have adopted as part of their mission the computer-related education of non-computer science majors. Service courses offered by the Computer Science Department are tailored to varying extents to satisfy the needs of particular groups of students--social science, natural science, humanities, education, and business.

Professional Schools and Academic Departments

Many of the professional schools and some academic departments require their student majors to take introductory courses in computer programming and applications. Such courses are designed to provide the student with skills he needs to pursue further course work in the degree program. At Colorado School of Mines, for example, all students have been required to learn programming in their freshman year. At Auburn University, such requirements affect all students in the Schools of Business, Education, Pharmacy, Engineering, Architecture, and Agriculture, as well as the Departments of Physics and Mathematics. These requirements affect 68% of the total student body at Auburn.

At some institutions, the introductory computer courses offered by the professional schools have been consolidated into service courses provided by the Computer Science Department. At the University of Texas, El Paso, for example, computer-related courses were first offered in 1966 by the Departments of Mathematics, Business, Mechanical Engineering, Electrical and Civil Engineering. Several of these courses have now been consolidated into a new computer science curriculum. The purpose of such consolidations is usually to eliminate duplications and to reduce costs.

Educational Research and Development Organizations

In the 1960's, projects and organizations were formed on some campuses to research the possibilities of using the computer as a medium, as opposed to an object, of instruction. Although the original intent was not to further the cause of students' computer literacy, this work has had a major effect in that direction. For example, research was started in 1960 in the Coordinated Science Laboratory at the University of Illinois to explore the possibilities of automation in individual instruction. By 1977, seventeen thousand University of Illinois students a year became familiar with computer programming, computer applications, and computer impact, via PLATO's varied programs of instruction, communication, computation and recreation.

Similar organizations at Fairfield University in Connecticut, and at Bennett, a small minority institution in North Carolina, have provided the leadership in introducing instructional computing on campus.

ACCESS TO COMPUTING FACILITIES

Student and faculty access to appropriate computing facilities is a necessary condition for computer literacy. Several ways of measuring the quality and quantity of academic computing facilities are suggested in Mossman (1). I will highlight here only those characteristics of computing facilities that have been cited as key ingredients in computing literacy by the exemplar institutions.

The installation of a time-sharing computer on campus is considered the significant event that made computer literacy a feasible goal at several exemplar schools. In this connection, several of the schools have adopted what has become known as the "Dartmouth model" of campus computing. As articulated by T.E. Kurtz and A.K. Morton of Kiewit Computation Center, the ingredients of this model are:

(1) A highly reliable and secure time-sharing system that faculty and students can use with confidence and which operates on a predictable basis.
(2) A system that is easy to learn and use, which provides clean interfaces to languages that are easy to learn and use, and yet is general enough to grow with the increasingly sophisticated tastes of the students and faculty.
(3) An open-access mode of operation.

These and other aspects of the Dartmouth model are discussed in Luehrmann and Nevison (2) and Nevison (3). The University of Wisconsin at LaCrosse, Grinnell College, the University of Tennessee at Chattanooga, Evergreen State College, and Colorado School of Mines have cited elements of this model as key ingredients in their computer literacy programs.

The number of interactive terminals available for student use has been cited as another key ingredient. At the exemplar schools, the student/terminal ratio ranges from fourteen to four hundred. At the U.S. Naval Academy, 300 terminals are available at 50 locations for use by 4300 students. Among the eight exemplar universities with over 8,000 enrollment, the average student/terminal ratio in 1976-77 was 217. Among thirteen exemplar schools with enrollment under 8,000, the average student/terminal ratio in the same year was 78. Nearly all the schools have student terminals at many locations on campus, and some have portable terminals that students can take home or to their dormitories.

The most commonly used language for computer literacy purposes at the exemplar schools is the language BASIC. BASIC is interactive and easy to learn. APL is also advocated for literacy purposes, as well as for computer science majors. At Fairfield University, APL was adopted as the official language in 1966. It is the major vehicle for introducing students to computers, the major language used by students for performing their classroom or laboratory work, and the major instructional language used by members of the Educational Research and Development Division staff and by members of the university faculty.

COURSES

For convenience in discussion, I would like to make a distinction between two kinds of courses. Papers presented in the annual Conferences on Computers in the Undergraduate Curriculum provide many examples of the first kind of course. The primary focus of such courses is to teach physics or chemistry or sociology, but the students also acquire skills and knowledges in computing. They learn about computer applications in their field of study, and they learn to become more proficient computer users. At most of our exemplar institutions, we find a very broad spectrum of such computer applications. Courses using computers are found in over 50% of the academic departments of these colleges and universities. The percentage of courses using computers ranges from 6% to 50%.

Courses of the second kind are more explicitly and primarily addressed towards computer literacy skills and knowledge. They include courses in "computers in society" and a variety of introductory computing courses.

Computers in Society

Nearly all exemplar institutions offer a course in "Computers in Society" or "Computers in the Modern World." These survey courses usually are addressed to lower division students who do not have prior computing experience. They provide a broad introduction to computer applications in engineering, science, business, humanities and education. Some include elementary use of the computer and an interactive language (APL or BASIC). Discussion topics include:

- privacy
- human obsolescence
- artificial intelligence
- effects on education
- computer control of systems such as monetary flow and national defense
- effects on individuality

A more intensive "contract" course developed at Evergreen State College by Dr. Jim Johnson includes analysis of current legislation regarding the use of computers. In addition, students are expected to research a particular computer system and analyze how it affects people: customers, users, clerks, computer personnel, managers, etc.

General Introduction to Computer Science or Programming

Commonly, the exemplar schools offer a general introduction to computer science. Students with varied backgrounds and interests take these courses. At Denison, for example, 50% of all students on campus take the Introductory Computer Science course. This course presents a general introduction to historical and technological aspects of computing systems, and teaches the rudiments of algorithms and programming. Students write and execute programs in BASIC, in a simple assembly language, and in a simple machine language.

It is difficult to satisfy the diverse needs and interests of a wide range of student majors in one introductory course. At Grinnell, the introductory course offers problem sets in different application areas so that individual students can work on problems appropriate to their major field of interest.

Computing with Scientific and Engineering Applications

At the Colorado School of Mines, all freshmen take Basic Engineering 101 Computer Science I. This course is the foundation on which all further applications of computer use and instruction are built. The students spend from 3 to 9 hours per week for one semester in the course which provides an introduction to computer programming, computing procedures, and the Colorado School of Mines computer system. Elements of FORTRAN, program planning, flow charts, and problem organization are applied to elementary engineering problems.

At the U.S. Naval Academy, the course "FORTRAN with Scientific and Engineering Applications" is intended to meet the computing literacy needs of scientific and engineering majors. The course includes problems dealing with physics, mathematics, probabilities and statistics, and various engineering applications.

An analogous course at Trinity University places special emphasis on the use of the APL language as a notation for problem solving in the natural sciences.

Introduction to Computing in the Social Sciences

Students majoring in social sciences do not as readily find introductory computing courses tailored to their interests. Beginning courses on the use of SPSS (Statistical Program for Social Sciences) are offered by some of the computer centers, such as the University of Texas at Austin.

At Dartmouth, the need for extensive training in the use of such social science programs is reduced considerably through the use of IMPRESS. IMPRESS is an interactive system that provides access to analysis techniques and a variety of large social data bases.

Introduction to Computing in Humanities and Fine Arts

The Computer Science Department at Bucknell University offers a course with applications of computing in fields such as art, music, literature, political science, and history. The students learn text processing, problem solving, and programming.

The Evergreen State College extends computer use to the arts through sponsorship of computer film festivals and encouragement of computer graphics. Students have made computer films which have been exhibited nationally.

Introduction to Computing in Education

Students at the University of Wisconsin, LaCrosse, who are studying to be teachers become knowledgeable users of computing via courses taught in the Secondary Education Department. Computer applications are integrated into many of the education courses offered. One introductory course, begun in 1972, teaches computer applications in instruction including computer fundamentals, BASIC programming, simulations, and elements of computer-assisted instruction. Another addresses the development of instructional and curriculum materials suitable for computer applications.

One component of the literacy program at Fairfield University is the introductory courses offered by the Education Department. "Introduction to Computers" is taught by computer in an interactive tutorial mode. "Computer Programming for Teachers" enables the student to learn to program in the APL language, and write instructional programs. A third course emphasizes the effective use of the computer in the classroom.

Mankato State University offers a Computer Science Teaching Degree Program designed for individuals who desire to use, teach, or coordinate the computer science curriculum of a junior or senior high school. There is a close alliance between the Computer Science and Education Departments, and a number of courses are offered on computers in education. Course titles include: "Computers in the Classroom," "Elements and Methods of Computer-Assisted Instruction," and "Administrative Uses of the Computer in Education."

Introductory Data Processing for Business Students

Exemplar institutions awarding a Bachelor or graduate degree in business usually require student majors to take an introductory course in data processing offered either by the Computer Science or the Business Department.

Since 1966, a three-hour course in business data processing has been required of all students en route to a BBA degree at the University of Texas at Austin. Approximately 2500 undergraduate students enroll in this course each year. The primary objective is to provide students with a fundamental knowledge of the interactions of computer hardware, software, and personnel in a business environment.

IMPACT

Student Users

One indicator of the impact of computer literacy programs is the proportion of students using the computer in their projects or coursework. Different institutions maintain different kinds of computer usage records, so that it is difficult to present precise and comparable information on this matter. However, each of the exemplar institutions has provided as accurate an estimate as possible of the numbers of students using computing in the 1976-77 academic year. This total, divided by the enrollment, gives a rough indication of the extent to which computer use pervades the campus.

Fifteen exemplar schools under 8,000 student enrollment ranged from 29% to 100% student users in 1976-77. The average proportion of student users was 57% for these smaller institutions. For seven institutions with 12,000 enrollment or larger, the range was 16% to 51% student users. The average for these seven institutions was 34% student users.

Increases in Student Use of Computers

At most colleges and universities, computer literacy has grown slowly, as various facilities and curriculum changes were added over the years. However, two of our exemplars provide examples of how quickly computer literacy can spread, once a full commitment is made by the institution.

Grinnell College had just a few student computer users prior to 1975, when their time-sharing system was installed on campus and computer literacy courses were offered. In one year, the number of student computer users jumped to five or six hundred out of a total enrollment of 1200. Similarly, at the University of Tennessee, Chattanooga, an on-campus time-sharing computer was installed in the summer of 1975 and the Office of Academic Computing was established. In the 1975-76 academic year, 75 faculty and 2050 students participated and were assigned account numbers. Previously, fewer than 100 students were using computing.

Surveys of Students or Faculty

Another method for assessing the impact of computer literacy programs is through surveys of students and/or faculty. The Computer Center at Carleton College has conducted such surveys since 1973. The Center uses these survey results to evaluate various components of the computer facilities and literacy programs. The survey is also useful in determining the need for introductory programs, since it asks about the computer experience of incoming freshmen. As one would expect, the incoming students are increasingly computer-literate. In the graduating class of 1980, 56% have already had exposure to computers before Carleton, and 43% had "intensive" use of computers before Carleton.

Surveys of student computer use at Dartmouth are reported in Nevison (4).

SUMMARY

A variety of approaches have been taken by the exemplar institutions to encourage and enable college students to become literate in computing. Based on their experience, we can identify key elements in these approaches:

(1) institutional commitment to academic computing and administrative leadership;
(2) student access to suitable computing facilities;
(3) academic computing services staff to provide training and assistance to faculty and students;
(4) computer center short courses and seminars;
(5) introductory courses in computing offered to all students by the Computer Science Department;
(6) introductory courses in computing tailored to needs of particular groups of students;
(7) educational research and development, including development of curricular materials and instructional methods;
(8) computer-based learning materials including data bases, program libraries, laboratories, problem sets and course outlines, etc.;
(9) incorporation and integration of computer-based learning into the regular courses of the various academic departments and student projects.

REFERENCES

(1) MOSSMAN, Charles. Evaluating Instructional Computing. Irvine: University of California, 1976.

(2) LUEHRMANN, Arthur W., and Nevison, John M. Computer Use Under a Free-Access Policy. Science, Vol. 184, No. 4140. May 1974.

(3) NEVISON, John M. Computing in the Liberal Arts College. Science, Vol. 194, No. 4263. October 1976.

(4) NEVISON, John M. Computing as a Matter of Course. Dartmouth College, 1976.

COMPUTER-ASSISTED FORECASTING OF ECONOMIC AGGREGATES

Ronald A. Krieger

Goucher College

I. PURPOSE

One of the major activities engaged in by economists in government and business is the forecasting of future economic conditions. Yet, the skills required for forecasting are seldom taught directly. Undergraduates rarely learn the techniques of econometrics used by forecasters; although economics majors typically are required to take a course in statistics, they seldom get deeply involved with multiple regression techniques. Nor is it common for them to learn to analyze and monitor business conditions in the light of business-cycle theory and history. Even graduate students generally fail to encounter this type of work, since their study of econometrics is normally confined to econometric theory until they begin work on their doctoral dissertations.

At Goucher College, a small liberal arts college for women, we have learned that undergraduates, aided by a time-sharing computer system (also available in batch processing) can acquire a sophisticated, if intuitive, "user's" knowledge of forecasting techniques. This is not just for the exceptional or the research-oriented students. In fact, all senior economics majors at Goucher learn to use these techniques in the one-semester Senior Seminar in Macroeconomics required for their degree.

II. BACKGROUND INSTRUCTION

The prerequisite to the course is a semester of intermediate macro theory and a semester of introductory statistics. In the first weeks of the course, students acquire two types of additional background:

A. Economic:[1] Through lectures and textbooks, they study time-series analysis, breaking down cyclical, seasonal, trend, and random components. They learn some business cycle theory and the history of post-war fluctuations in economic activity. They also study the reference-cycle method developed by the National Bureau of Economic Research for dating peaks and troughs.

B. Computer: Using remote terminals tied in with the DECSYSTEM-10 equipment at the Johns Hopkins University Computing Center, class members are given individual instruction in the essentials of timesharing on that system. They learn how to create, edit, and manipulate files using DEC's SOS editing routine and practicing on a few simple programs in BASIC. They then learn to create and run

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

regression programs using the Time Series Processor (TSP) language developed at Harvard and the Massachusetts Institute of Technology.[2] This is a simple computer language for the statistical analysis of time series using ordinary least squares and two-stage least squares. The programs are short, and students need have no previous programming background.

III. PRELIMINARY RESEARCH

The stage is now set for each student to undertake an individual forecasting project, picking a variable representing some aspect of the aggregate U.S. economy (see Appendix A for a list of project variables used in 1978 forecasts). A typical variable is the Consumer Price Index. The student puts together her own bibliography of books and journal articles that will allow her to study the theory and cyclical history of price fluctuations. Based on her research into inflation theory, she identifies a number of independent variables that might help to explain variations in the rate of inflation. The instructor assists her in final identification of the appropriate data series that represent each variable. She then turns to such sources as the Business Conditions Digest, Federal Reserve Bulletin, Survey of Current Business, and International Financial Statistics to gather quarterly data for the past 25 years on dependent and independent variables she has identified. She then keypunches the data according to a prescribed format and enters the data into disk storage via card reader, to be accessed through remote terminals by TSP programs that she will also store on disk.

Before proceeding with the empirical work, the student at this stage writes a rough draft of two chapters of her final research paper: one dealing with the historic cyclical behavior of her dependent variable, to assist her in forecasting, and the other containing her review of the theoretical literature.

Meanwhile, lectures and reading are building up an intuitive understanding of multiple regression techniques and the associated statistics.[3] Through a previous statistics course, the students already have encountered statistical inference and simple regression. They are now given a non-mathematical understanding of multiple regression results, including the interpretation of coefficients, R^2, standard errors, t-ratios, and F-statistics. They learn how to handle problems of serial correlation identified by the Durbin-Watson statistic and how to deal with multicolinearity as indicated by regression results and a correlation matrix. They are also taught the uses of data transformation, especially the log-linear and delta-log forms. Finally, they learn to use both discrete lags and the polynomial distributed lag.

IV. ECONOMETRIC WORK

Armed with this background, the students now begin a "hands-on" exercise that will reinforce their knowledge of econometrics. They spend from four to six weeks at the computer terminals trying to estimate the parameters of a single-equation model (in spirit, a reduced-form model) based on their theoretical research. The only technique used is Ordinary Least Squares, which rules out the use of lagged endogenous variables as regressors. They write a simple program in TSP (see Appendix B), creating it on the terminal using the SOS editor, and try an initial run regressing the untransformed dependent variable on all the independent variables. The first run usually includes a printout of all the data and a correlation matrix for all the variables in the equation. It provides a useful check on the accuracy of the keypunching, and suggests to the students the directions that their experimentation might take. Checking the signs of the coefficients, t-statistics, R^2, Durbin-Watson, and individual correlations, they learn, through consultation with the instructor, to interpret these results and decide what to do next. Each successive run requires them to edit their TSP programs in SOS, putting in and taking out data transformations, lag distributions, and variables, often running several regressions at a time. Using the Almon lag, they experiment with the degree of the polynomial, length of the lag, and endpoint restrictions. Eventually, every student arrives at a final result which is satisfactory to a greater or lesser degree, given the six-week time constraint for the research. The equation they are hoping for is one suggested by theory, with all coefficients significant and of the right sign, a high coefficient of determination, and an absence of serial correlation of the residuals (see Appendix C for a typical final result).

At this point, the students write up the empirical chapter of their research paper, describing the general approaches used, interpreting the final equation, and relating the empirical results to the theoretical suppositions of the previous chapter.

V. FORECASTING

The next step is the development of a conditional forecast using the regression equation, trying out various scenarios for the independent variables. During the semester the class members have been closely following such sources as the Wall Street Journal and Business Week to acquire a feel for current economic conditions and future government policies. This suggests to them a series of values for the independent variables for the next four calendar quarters, and they use the TSP program to produce a mechanical forecast for that period. They insert commands into their programs to retrieve the coefficients calculated for the sample period, then compute a vector of fitted values for the forecast period. Since the point has been made repeatedly throughout the course that the regression equation allows only a consistent extrapolation of the past, they are all required to supplement their mechanical forecast with a judgment forecast based on their knowledge of economic conditions and the past cyclical behavior of their dependent variable.

The final class exercise is the construction of small simultaneous-equation models for various sectors of the economy, with students putting their own work together with the work of other students. They test each equation to determine whether it is identified, and estimate each identified or overidentified equation through two-stage least squares, using a simple modification of their TSP programs.

VI. CONCLUSIONS

Without exception, every student majoring in economics at Goucher in the past five years has managed to get through the econometric phase of this exercise and enter the world beyond the campus with a good intuitive understanding of forecasting procedures and technique. At the end of the term, each student undergoes a final oral defense of her paper, which tests her understanding of the statistical methods used and the results given in each computer printout. All have passed the exam, owing their success more to the learning acquired by working on the computer than to the book-and lecture-instruction. The latter aspect is necessary but clearly insufficient at the undergraduate level. The success of the forecasting can be measured against the actual performance of the economy, and Goucher students generally do as well as or better than the consensus forecast of the National Association of Business Economists. More fundamentally, recent graduates report that the techniques acquired in this seminar have proved indispensable to them on the job in businesses, banks, and brokerages, or in graduate studies in economics or business, where subsequent formal study of econometric theory has been informed and enhanced by their practical econometric experience as undergraduates.

FOOTNOTES

---

[1] The principal textbook used for this portion of the course is Carl A. Dauten and Lloyd M. Valentine, Business Cycles and Forecasting, 4th ed. (Cincinnati: Southwestern Publishing Co., 1974).

[2] Documentation available as Technical Paper No. 12 of the Harvard Institute of Economic Research, Cambridge, Massachusetts, Bronwyn H. Hall, TSP: Time Series Processor, Version 2, revised August 1976. This was developed for the IBM 370/168 at MIT and adapted to the DEC-10 by the Johns Hopkins University Computing Center for both batch and time-sharing.

[3] A nontechnical text is used in this part of the course, Roger K. Chisholm and Gilbert R. Whitaker Jr., Forecasting Methods (Homewood, Ill.: Richard D. Irwin, Inc., 1971).

APPENDICES

A. List of research projects used by 31 students in the fall 1977 seminar, as targets of quarterly forecasts for 1978. Numbers in parentheses identify appropriate data series in the Business Conditions Digest (old series numbers).

B. Final program developed by student in 1977 seminar to forecast consumer prices for 1978, and the regression output that resulted.

C. The final regression result as presented by the student in her seminar paper.

Economics 395 RESEARCH PROJECTS Fall 1977

AGGREGATE DEMAND

Personal Consumption Expenditures

1. Consumer nondurables (236)
2. Services (237)
3. Consumer Durables (233)
4. Automobiles (234)

Gross Private Domestic Investment

5. Capital spending (241)
6. Housing (244)
7. Change in business inventories (245)

Government Purchases of Goods and Services

8. Federal (262)
9. State and local (266)

Net Foreign Investment

10. Exports (252)
11. Imports (253)

AGGREGATE SUPPLY

12. Employment (48)
13. Unemployment (43)
14. Personal saving (292)
15. Productivity (770)
16. Profits (16)
17. Industrial production (47)
18. Federal tax receipts (601)
19. Capacity utilization (850)
20. Wages (740)

MONETARY VARIABLES

21. Short-term interest rates (114)
22. Long-term interest rates (116)
23. Bank lending to businesses (72)
24. Consumer instalment debt (66)
25. Basic money supply, M1 (85)
26. Broad money supply, M2 (102)
27. Extended money supply, M3 (103)

PRICE INDICES

28. Consumer prices (781)
29. Wholesale prices (750)
30. GNP deflator (210)
31. Stock prices (19)

APPENDIX A

```
      TIME SERIES PROCESSOR VERSION 2.7A FOR THE DECSYSTEM-
 14-Dec-77    19:13   NONAM        PAGE    1



    1.  $$NAME FANG10 'FORECAST'$

                                                 PROGRAM

  LINE  *******************************************************
****************

    1.  $$NAME FANG10 'FORECAST'$

    1. -LOAD$

    2.  SMPL 1 100$

    3.  READ (25,100) CPI M1 M2 WAGE PROD FBP$

    4. -100 FORMAT '(6F10.0)'$

    4.  SUPRES 15 17 42$

    5.  GENR DLNCPI = LOG(CPI) - LOG(CPI(-1))$

    6   GENR DLNM1 = LOG(M1) - LOG(M1(-1))$

    7.  GENR DLNW = LOG(WAGE) - LOG(WAGE(-1))$

    8.  GENR DLNW1 = DLNW(-1)$

    9.-GENR DLNP = LOG(PROD) - LOG(PROD(-1))$

   10.  GENR DLNP1 = DLNP(-1)$

   11.  GENR FBP1 = FBP(-1)$

   12. -SMPL 39 94$

   13.  PDL DLNCPI C DLNW1 DLNP1 FBP1 DLNM1(3 17 3)$

   14.  RETRV COEFF 23 V$

   15.  SMPL 39 100$

   16.  PDL DLNCPI C DLNW1 DLNP1 FBP1 DLNM1(3 17 3)$

   17.  SMPL 85 100$

   18.  FIT COEFF YHAT C DLNW1 DLNP1 FBP1 A00001 A00002$

   19.  PRINT DLNCPI YHAT C DLNW1 DLNP1 FBP1 A00001 A00002$

   20.  SMPL 1 100$

   21.--END$
```

APPENDIX B.1

```
EQUATION   1
************

ORDINARY LEAST SQUARES

DEPENDENT VARIABLE:   DLNCPI

      RIGHT-HAND                   ESTIMATED                STANDARD                T-
       VARIABLE                   COEFFICIENT               ERROR               STATISTIC

        C                       -0.100799E-01           0.248234E-02            -4.06065
        DLNW1                    0.407810               0.100556                 4.05554
        DLNP1                   -0.175441               0.725320E-01            -2.41881
        FBP1                    -0.557442E-04           0.370676E-04            -1.50385
        A00001                   0.808687               0.150962                 5.35687
        A00002                  -1.02401                0.213330                -4.80010

R-SQUARED =  0.7415
DURBIN-WATSON STATISTIC (ADJ. FOR 0. GAPS) =  1.6957
STANDARD ERROR OF THE REGRESSION =        0.386440E-02
SUM OF RESIDUALS =        0.143191E-07
NUMBER OF OBSERVATIONS =  56.
MEAN OF DEPENDENT VARIABLE =        0.121713E-01
F-STATISTIC(  5.,  50.) =        28.6778
```

```
                  DISTRIBUTED LAG INTERPRETATION

   COEFFICIENT ST'D. ERROR T-STATISTIC PLOT OF THE LAG DISTRIBUTION(*) AND ST'D. ERROR BAND(+)

 0  0.4177E-01  0.7739E-02   5.397                                  .    +*+
 1  0.7721E-01  0.1418E-01   5.443                                  .        + * +
 2  0.1063      0.1934E-01   5.497                                  .           +  *  +
 3  0.1291      0.2323E-01   5.559                                  .              +  *   +
 4  0.1456      0.2586E-01   5.632                                  .                +   *  +
 5  0.1558      0.2725E-01   5.717                                  .                 +   *   +
 6  0.1596      0.2745E-01   5.816                                  .                  +   *   +
 7  0.1571      0.2652E-01   5.925                                  .                  +  *   +
 8  0.1483      0.2461E-01   6.029                                  .                 +  *   +
 9  0.1332      0.2197E-01   6.065                                  .               +  *  +
10  0.1118      0.1919E-01   5.826                                  .            +  *  +
11  0.8401E-01  0.1750E-01   4.801                                  .         + * +
12  0.4992E-01  0.1874E-01   2.664                                  .    + *  +
13  0.9518E-02  0.2383E-01  0.3994                                + . *  +
14 -0.3721E-01  0.3211E-01  -1.159                         +   *    +
15 -0.9026E-01  0.4274E-01  -2.112                +     *     +     .
16 -0.1496      0.5523E-01  -2.709      +     *        +            .

                  MEAN LAG =         4.03951
                  STANDARD ERROR =         1.85055
                  SUM OF LAG COEFFICIENTS =         1.23232
                  STANDARD ERROR =        0.233820
```

APPENDIX B.3

The final equation is:

$$\hat{Y} = -0.01 + 0.408X_1 - 0.175X_2 - 0.00006X_3 + \sum b_i X_{4,t-i}$$

(4.10) (-2.42) (-1.50)

(t-statistics in parentheses) $R^2 = .74$ DW = 1.70

Second-degree polynomial lag coefficients:

$b_0 = 0.04$ (5.40)
$b_1 = 0.08$ (5.44)
$b_2 = 0.11$ (5.50)
$b_3 = 0.13$ (5.56)
$b_4 = 0.15$ (5.63)
$b_5 = 0.16$ (5.72)

$b_6 = 0.16$ (5.82)
$b_7 = 0.16$ (5.93)
$b_8 = 0.15$ (6.03)
$b_9 = 0.13$ (6.07)
$b_{10} = 0.11$ (5.83)
$b_{11} = 0.08$ (4.80)

$b_{12} = 0.05$ (2.66)
$b_{13} = 0.01$ (0.40)
$b_{14} = -0.04$ (-1.16)
$b_{15} = -0.09$ (-2.11)
$b_{16} = -0.15$ (-2.71)
$\sum b_i = 1.23$ (5.27)

Where:

Y = delta log of the Consumer Price Index for all items, 1967 = 100.
X_1 = delta log, lagged one quarter, of average hourly compensation, all employees, nonfarm business sector, 1967 = 100.
X_2 = delta log, lagged one quarter, of output per hour, all persons, total private economy, 1967 = 100,
X_3 = the Federal Budget Position, lagged one quarter, in billions of current dollars, annualized (- if deficit, + if surplus).
X_4 = delta log of M1, originally in billions of current dollars.

Sample period: quarterly data from 1963 III through 1977 III.

Making Large Classes Work in Computing for Business

Thomas Rigo and John M. Gersting, Jr.

Department of Mathematical Sciences, Computer Science Group
Indiana University - Purdue University at Indianapolis
Indianapolis, IN 46205

BACKGROUND:

In 1973 the School of Business at IUPUI requested the Computer Science group of the Department of Mathematical Sciences to design and implement a course to introduce sophomore business majors to digital computers and their use in business. At that time the staff resources that could be devoted to such a course were small and no teaching assistants were available. Since the expected enrollment was 500 to 600 students per year, it was relatively clear from the beginning that the only way the course could be offered to service the projected student load was by concentrating the students in a few large sections.

What follows is a description of the developments over the past 5 years which we think have turned a usual disadvantage, large class size in a computing course, into an advantage and has helped shape the course we call CSCI 208 -- The Computer in Business.

COURSE OBJECTIVES:

CSCI 208 is a required course for sophomore business majors and is the only CS course required of these students. A prerequisite of intermediate algebra is assumed (students eventually must take finite mathematics, calculus for business students and accounting). Two major objectives of the course are:

1) to expose the student to terminology and fundamentals of data processing, and
2) to help the student learn to program a computer at an elementary level using the FORTRAN language (FORTRAN is chosen in preparation for later computer oriented courses in the business curriculum).

COURSE CONTENT:

There are both programming and discussion aspects to the course. The programming portion of the course begins with 10 statement FORTRAN in a batch processing mode on a CDC 6600 and expands to coverage of most of ANSI.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

FORTRAN including subroutines and I/O statements for file manipulation running in a time sharing mode on a DEC-10. Four to six FORTRAN programming exercises are assigned during a typical semester. COBOL implementations of some of these exercises are studied by the students but no COBOL coding is required.

The discussion portion of the course is divided into two parts, a systems analysis section and an overview section. The major emphasis in the systems analysis section is placed on flowcharting and the use of simple control structures to decompose a problem into manageable units. These units together with the concept of a subalgorithm or subroutine lead to a modular approach to programming. The overview section includes discussion of hardware components of a computer, supporting equipment such as the keypunch and terminal, storage media such as tape and disk files, and brief discussions of mini, micro and hand held computers. Typical business applications of computing are also examined.

TECHNIQUE:

Introduction

For the most part the stated objectives and content could fit any computing course. The question is how to cover all the material in 45 class hours (3 credit hours) using one instructor and a part-time grader with a class of over 100 students? Two points became obvious quickly: the instructor could spend a considerable amount of his time simply administering the course, and if programming was to be required of the students, making the same assignment to each student could put considerable pressure on the students to "help" each other, "buy/sell" code, or otherwise complete assignments without actually doing the work. To attempt to alleviate the pressure on the instructor (to allow more time to answer student questions) and the pressure on the students (to help or seek help of each other) it was decided to fully computerize the administration of the course, including record keeping, assignment generation and collection, and examination generation and grading.

The task of designing and implementing the computerized support system was lengthy and stretched well into the first semester the course was offered. The initial class got a day by day report of what was happening. We completed the package and the first class survived the course.

The second semester we just used the system but student involvement in the class seemed to wane and we couldn't seem to cover as much as we wanted. Then we realized that the first two semesters differed greatly in their environment, even though the content was seemingly the same. In the first semester the students were involved in the development of the support system, in the second they were uninvolved bystanders.

The conclusion we came to was that to get the maximum effect from the course we should involve the student in the course as much as possible. Since the course is for business majors and since most of these students will eventually be exposed primarily to the environment of business data processing and not heavily involved with syntax of a language or details of implementation of algorithms, we decided that establishment of a DP environment was important pedagogically as well as administratively. We decided to use the administration of the class itself as an example of a data processing system, and to plunge the student into this environment in every conceivable manner. The student would then play the role of a user as well as a programmer: as a programmer, he must produce working code to satisfy some superior (the instructor); as a user, he must learn to interface with and utilize the various programs which provide administrative support functions for the class.

The Student "N" Number

The DP environment established around the course is designed to be precise (fixed format in batch applications, somewhat more flexible formats in time sharing applications) and is based on a unique number assigned to each student. This identifying number, called the student "N" number, is assigned in the first class. It is used throughout the semester to facilitate everything from recording of grades to generation of program assignments and exams.

After a lecture/demonstration on the use of a keypunch the first "homework" assignment given to the students is to keypunch a supply of "N" cards which contain the student's name and number in a prescribed format. The students then see their first "batch application"; while attendance is not required, students present at a lecture are asked to hand in an "N" card, this "input" is then listed and posted in the next class (it only takes about 3 classes to get all the "N" cards into the correct format). If no "N" cards are received from a particular student for 5 consecutive classes the system "deactivates" the student and no additional assignments are generated for him.

Further uses of the student "N" number are illustrated in the sections that follow.

Programming Exercises

Although the purpose of the programming exercises is to increase the students' ability to take an English language problem statement, perform appropriate systems analysis, and implement the resulting algorithms using a higher level language, the generation of the program write-ups and the collection and testing of the resulting code are all part of the DP environment in which the student plays the role of a user.

In particular, each programming exercise is designed with a number of different assignments (usually 2 to 4) involving differing algorithms and a number of options within each assignment (usually no less than 4 and often as many as 20) including items such as differing formats or program constants. The selection process is carried out using the student "N" and the modulo (MOD) function.

As an example consider a typical programming exercise concerned with reading card input, simple computations, totalling, and output of a printed table. An often used topic for such an exercise is a GNP tabulation. The write-up begins with a short business oriented discussion explaining the GNP, some of its uses and a formula for computation of the GNP. Two formulas are used: product flow, assignment #0, and cost flow, assignment #1. A section on algorithms presents the idea of a base year and conversion to a new base year using a deflator. Each assignment contains 5 options (numbered 0 thru 4) where a base year and deflator value are given along with a set of data cards. Each data set is in a different format and contains a year and the necessary values for quantities in the GNP formula used in that assignment (e.g., C, I, G, XN, D for product flow). All input data is in 1958 dollars. Output of the programs is to consist of printing the input, NNP, GNP, and GNP in the new base year. Finally the number of cards processed is to be output. A computer oriented discussion in the write-up makes suggestions on how items such as the "loop" can be implemented and how the test for end of data can be made.

The actual assignment might be made thru the following computations

assignment # = MOD(N,2)

option # = MOD(N,5) for both new base year and data set

For the particular case of the student whose "N" is 247, he would get assignment # = MOD(247,2) = 1 or the cost flow assignment

option # = MOD(247,5) = 2 or a new base year of say 1933
and data set #2.

Using this process 10 distinct programming exercises are generated. If more were desired one could use, say, 6 data sets and MOD(N,6) for a data set number, then with MOD(N,2) for assignment #, MOD(N,5) for new base year, and MOD(N,6) for the data set # 60 distinct exercises would result.

Two different modes of publishing program exercises have been used. When course notes [1] are used all assignments are preprinted, then the MOD function for the assignment # is not given until the day of the assignment. More recently software has been developed to generate customized assignments on the computer for each active student. This software uses text segments stored on random access files and MOD computations to produce an assignment with the appropriate option built right in. This reduces the complexity and length of the write-ups since there need not be large tables containing options and an individual student need only receive the write-up for the assignment he is going to implement, e,g., for the GNP example the student would get the cost flow material with the new base year of 1933 and its deflator value inserted in the proper place in the text.

Some additional topics used for program exercises are given below.

Topic	Assignments	Options
DP -- merge	weekly payroll payroll deductions back pay adjustment stock level	overtime factor FICA limit overtime factor margin
Location of distribution center	Indiana map grid midwest map grid	cost/ton cost/ton
Inventory level simulation	3 different arrival distributions	random number seed

There are some other items we found helpful in creating new assignments. Something should be included besides input formats that differs in each option (e.g., the new base year in GNP). This causes internal coding differences in the programs as well as different input formats. Also one can bias the MOD functions by using MOD(N+3,5) or MOD(N,5) + 3 to obtain interesting variations.

We think the seeming complexity is worth the effort since it allows friends who come to class together, hence get consecutive "N" numbers in the first class, to study together without doing each others homework since the assignment algorithms will guarantee consecutive numbers functionally different programs. All assignments are fundamentally the same algorithm but disguised by the write-up. When students start to see this it is a good indication that you are getting the basics thru to them.

Collection and Grading of Programming Exercises

We collect only code from the student and rerun all the programs against a standard data set (in the GNP example the data sets in the various options are designed with great care to provide "common" portions of all fields into which we place the standard data values). Results are checked by the grader against the standard data run thru master programs that produce all the necessary MOD computations. A program that fails to compile or abnormally terminates in execution is given a fixed grade of 3 out of a possible 20. No shows get a zero, one zero and you fail the course. Late points accrue at the rate of -2 per day with a minimum grade of 4 (or 3 if it "bombs").

In the batch portion of the course we collect and rerun decks. We hang on to batch since we feel everyone should have to see a "card" at least once. In time sharing we use a part of our "environmental support system" called TURNIN to collect and reexecute programs submitted for a grade. TURNIN is an interactive FORTRAN program that accepts information about the student's identity ("N", name and PW) and the name of the source code file(s) to be submitted for a grade. TURNIN carries out all the details required to place a copy of the source code file(s), standard data set(s), and control cards into the DEC-10 batch queue for execution. The DEC-10 has facilities whereby even interactive jobs can be executed in the batch queue. TURNIN then logs the time and date of submission, issues the student a receipt, and can set "flags" to prevent resubmission of that same program.

TURNIN and another program called GRADE to be discussed in the next section are FORTRAN programs that run on the DEC-10 and are portable except for the variations in the random access I/O implementation between machines. These programs are fully documented both from a users view [2] and from a technical view [3].

From the DP environmental view TURNIN and GRADE give us a chance to discuss the problems invloved with data base management and interactive access to a data base. We talk about the design of the records, the linking of the file, and privacy and protection considerations. We have found that giving the student the total responsibility for submitting a program for a grade under the control and guidance of the computer gives the student a keen awareness of the existence of the computer and some idea of what it can and sometimes doesn't do (only hearty operators can take the flack when the system crashes with 5 or 10 users running TURNIN).

Grade Reporting

Few things are closer to a student's heart than grades. In the CSCI 208 system design we tried to cope with questions like "What did I get on the last program?" by developing an interactive package to access the data base maintained for class record keeping. The program called GRADE has a batch application that takes in "N" cards with program or exam identifiers and creates a random access file with one record for each student keyed by the student "N". Each record contains the student's "N", name, a password for identification in the time sharing mode, grades for each item in the course, flags for various purposes, information supplied by TURNIN, times and dates of accesses and a message field which may be forced out to a student upon access to the data base (used to alert a student to problems in his data or simply "see your instructor").

The instructor can request various types of output from the data base including listings (raw data, alpha sort on name, sort on "N", ranked by %, by section, letter grades, etc.) and bar graphs of student performance. The student can assess his or her own record and display certain portions of it including grades for the various items, his average and the class average. The student may also request a histogram type output of a portion of the class around his average (marked by ↑, class average is *). A sample interactive session with GRADE follows, underlined material is entered by the student, *italic type is explanation.*

```
.RUN GRADE                                   execute the program
GRADE - 16:21  30-Nov-77
CLASS: CSCI208                               class identification
SECTION #: A735
STUDENT N: 247                               individual identification
NAME: JONES
```

```
PW: ______                                         password is not echoed
     INDIVIDUAL MESSAGE TO JONES                   message forced out to student
     SEE YOUR INSTRUCTOR ABOUT PROGRAM 5
E = END, N = NEWS, G = GRADES,C = COMMENT, H = HISTOGRAM
ENTER OPTION: G                                    display option is chosen

GRADING DATA AS OF 28-Nov-77                       student's averages
N = 247   NAME = JONES                             weighting factors
   TYPE      AVE       %    CLASS AVE&STDEV          GRADES

PROGRAMS  96.00    70.00    92.67  10.12    20  20  20  20  16
EXAM      46.67    30.00    59.75  11.46    20  10  12
  TOTALS  81.20   100.00    82.79   8.90        raw scores
                                                class totals
E = END, N = NEWS, G = GRADES, C = COMMENT, H = HISTOGRAM
ENTER OPTION: H                                 student totals

                                                histogram option is chosen
GRADING DATA AS OF 28-Nov-77

N = 247   NAME = JONES     + AVE = 81.20   * CLASS AVE = 82.79
                                          number of students in each 1%
                                          interval
#     0  2  7  6  4  7  5  8 14  9 11 20 10  8  2
%   74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89       * class ave.
                           +     *                        + student ave.
E = END, N = NEWS, G = GRADES, C = COMMENT, H = HISTOGRAM
ENTER OPTION: E

END OF EXECUTION
CPU TIME: 0.94  ELAPSED TIME 2:41.83
EXIT
```

The student can also list a news file for that class (the N option) or write a message into a message file (the C option) to alert the instructor to problems that exist in his records. The student has access to this program anytime the computer is running (almost 24 hours a day). The data base is updated after each item is graded.

Examinations

Examinations are also generated on a custom basis for each student. Usually they are simple permutations of question texts from a set of questions on a random access file. The exams are multiple choice questions designed to test the student's understanding of concepts and to test his ability to read someone else's FORTRAN code (writing ability is evaluated thru programming exercises). The exams are usually machine scored using a SCANTRON, scores are then transferred to "N" cards and added to the data base.

We have tried a fully automated testing process using an individualized response system installed in our lecture hall which has a selector at each student seat for selection of one of 5 choices. The responses are collected by a computer nearby and output on paper tape for item analysis. This process was too much, under exam pressure and with varying reading ability several students "cracked" and couldn't cope with the computer so we dropped the idea. Now the response system is used as a demonstration of the use of a dedicated computer system when we need to "vote" on something like the date of the next exam.

Lecture

CSCI 208 is taught in a tiered lecture hall seating 100. The room is equipped with closed circuit TV and a raised rear-projection screen for film and slides with an overhead projection screen directly below it. We make heavy use of the two screens. We developed a set of 300 35mm slides which we use in conjunction with a set of skeleton lecture notes [1]. These notes describe a particular topic, pose a question, and leave space for the student's answer. In lecture the answers are rear-projected on one screen so the students can compare them with their work or simply copy them. The overhead is used for discussion and/or variations on the answer. Some instructors use just the notes, others use a text, and some use parts of both.

The TV system is used mainly for demonstrations. While discussing the use and operation of the keypunch and the terminal the TV is used to help point out operating characteristics of these devices. Often we demonstrate software by connecting the TV system directly to a CRT terminal and displaying the terminal session on 6 large screens.

FINAL COMMENTS:

CSCI 208 is in its tenth semester and still is undergoing some change but we feel we are now able to meet the stated objectives. It is difficult to judge the persistance of the facts and concepts made available to the students beyond the end of the course. At the end of the course a large majority (80%) of the students completing the course (about 30% drop or stop attending which is about the same as other introductory computing courses at IUPUI) completed the code for a merge/sort problem consisting of about 300 FORTRAN statements including 2 subroutines and 3 disk files.

A small number of business majors eventually take upper division CS courses. They seem to be as knowledgeable and as successful as those students who complete the introductory programming course for science majors.

Whatever else might be said about the effectiveness of the methods employed in teaching CSCI 208, it certainly is true that the student is made aware of one of the fundamentals of data processing--the consistency of change. Experimentation with teaching techniques and audio visual aids, changing computer hardware and software, and occasional crisis management sessions have all contributed to an atmosphere not unlike that with which the student turned professional must cope.

ACKNOWLEDGEMENTS:

The authors wish to thank the 2500 or so students who have attended CSCI 208 over the past five years for their helpful suggestions and the IUPUI Computer Center, especially the consultants, who answer many questions for our students and the software staff who have helped us iron out many problems, in particular during the first offering of the course when the DEC-10 was being installed.

REFERENCES:

1. Gersting, J. M., "The Computer in Business - Notes for CSCI 208", IUPUI, 1973, 200 page monograph.
2. Rigo, T. and Gersting, J. M., "The GT Option for a Computing Class", Proceedings 3rd Annual Conference of Instructional Computing Applications, Indiana University Computing Network, at IU South Bend, March 1976, pages 169-183.
3. Gersting, J. M. and Rigo, T., "Documentation for the GRADE/TURNIN System", IUPUI, Computer Science Report TR 77-3, 1977, 71 pages.

DEVELOPMENT AND IMPLEMENTATION OF AN INTERACTIVE ACCOUNTING SYSTEM

Louis Geller, Chairman
Department of Accounting
and Information Systems

Queens College, CUNY
Flushing, New York 11367

That the computer has had an overwhelming influence over the way American business is managed and conducted has been both widely publicized and accepted as fact. There has been a development and application of hardware and software of varying degrees and levels of sophistication to decision making and transaction and data processing functions. Computer oriented business systems abound throughout the economic environment in which business is conducted. Integrated management information systems which have evolved from existing financial and managerial accounting systems are being installed and implemented in ever increasing numbers.

It may be assumed that the academic community which includes the necessary disciplines to provide both the research, developmental and technical facilities to support curriculum expansion to keep pace with the computer explosion has made its contributions.

However, academic associations, professional organizations, and the business community at large appear to be in agreement that education available to prospective accountants and auditors, as well as prospective computer consultants, is viewed ... as far more limited than it should be.

More specifically, recent university graduates have indicated that they have had inadequate preparation in the area of computer application as a result of the conventional emphasis in campus courses on "... a problem-solving mode, using canned packages or programming .... They seem to have very little feel for accounting transaction processing: the concepts of files, transaction updates, editing, reporting, and so forth. The second common deficiency was in flowcharting analysis and documentation, ... The third mentioned the area of auditing with a computer, including application of statistical sampling concepts, as well as the use of the computer in sample design, selection, and evaluation ... Many new auditors also lack the ability to visualize even a simple system or integrate the various EDP concepts they may have picked up. . . (using) university computers." [1] ". . . we are no longer graduating computer illiterates. However, just how computer-literate are our graduates?"[2]

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

DEVELOPMENT AT QUEENS COLLEGE

Phase 1 - Instructional Materials

It was in response to his awareness of the foregoing problems that the writer of this paper began in 1974 to work with small groups of students never numbering more than six, each consisting of a mix of computer science and accounting majors. The operation, still ongoing at the present time, is structured in the form of a tutorial dedicated to research in and development of interactive systems for applications as educational and accounting information and decision making tools. Participating students earn three credits per semester.

The first efforts during the spring of 1974 produced several modules written in Fortran which provided the group with the capability to successfully operate a perpetual inventory deliberately limited to five product lines at a wholesale level using an average cost calculation. After a review and discussion of directions to be pursued it was decided that the balance of the semester would be devoted to the design of a complete accounting system structured to meet the needs of a wholesale merchandising firm organized as a corporation. The system would be written in Fortran, would be sufficiently detailed to permit the operation of a complete set of accounting records (accounts receivable and payable were to be limited to a few accounts in the general ledger) including all necessary books of original entry. The availability of a general journal capable of processing adjusting and closing entries, routines to permit the preparation of interim as well as year end balance sheets and income statements, and a revision of the perpetual inventory system to provide for re-order and back-order reports at the input terminal while regular sales and purchase transactions were being processed, would meet the needs for a total information system.

All the design and developmental work during that first semester were performed on a Hazeltine terminal and a portable printing terminal both borrowed from a most cooperative computer center director and the Associate Dean of Faculty for the Social Sciences.

The resulting system (which is described in appendix following) evolved during several subsequent semesters and has been extensively modified and expanded by the writer. "Debit" is the file name assigned to the system.

The entire system including the necessary data files is stored on public disk at the Queens College Computer Center. The hardware is Xerox Sigma 7 using Xerox CP-V in time sharing mode. It may be noted here that the system has been written in Fortran IV and may be modified for use with other hardware and compilers with a minimum of problems.

By the Spring of 1977 the complete computerized accounting system previously described had been integrated into the first semester accounting course in conjunction with the textbook chapter on accounting systems and EDP. The writer functions as "visiting professor" in each of the sections offered at Queens College and provides a two hour lecture-demonstration using a portable terminal with video output to which are connected from four to eight 18 inch TV monitors depending upon the size of the classes taught.

Phase 2 - Equipment

The writer was awarded an H.E.W. grant for the acquisition of computer peripheral equipment during the fiscal year 1975-1976. Queens College installed multiple telephone jacks in three classrooms assigned to the Department of Accounting and Information Systems. Among the equipment acquired was a system partly designed by the writer, labelled "On-line classroom".

The system consists of a CRT terminal, with built in acoustic coupler cabled to a monitor, a control device with 15 switches and 15 cable outlets to which may be connected up to 15 auxiliary keyboards, with standard terminal configurations (ASCII) spread around the classroom which are used by students, and at the present time 4 eighteen inch screen video monitors placed strategically for viewing purposes, which are chained to a small monitor facing the instructor into which the terminal output is fed. In addition a printing terminal is interfaced with the instructor's terminal to permit hard copy to be prepared as the class progresses. The system permits the instructor to lecture and demonstrate on-line (phone hook-ups are made via the built-in acoustic coupler). Any one of the 15 switches may be set to the ON position permitting the student sitting at a particular keyboard to take over control and operation of the system. Only the instructor's terminal may be used to override a student keyboard when necessary. The multiple keyboard concept cuts the cost of the on-line operation because it permits the entire learning process to take place over one telephone extension, utilizing only one computer part.

Phase 3 - Applications

During the spring semester of 1976 the writer, as an experiment, undertook to teach several recitation-laboratory sections of the conventional Computer Science I course offered by the Department of Computer Science of Queens College. Students attend the regular lecture meetings - 2 hours per week - of the course in which FORTRAN is the programming language taught. Further the students were to be responsible for the usual assignments required of all Computer Science I students. However, registration in the lab sections to be taught by the writer was to be limited to accounting majors who had previously completed the first year of their accounting studies. These lab sections, meeting one hour per week, involved flow charting and programming accounting applications, and in addition were required to complete three projects designed by this writer.

Students in these sections were provided with introductions to the concepts of transactions processing, file structure and maintenance and actual hands on experience with the operation of the previously-mentioned interactive accounting system as well as batch mode processing.

Actual demonstrations and hands-on applications in class rooms equipped with telephone jacks have become a reality as the result of the acquisition of the computer-peripheral equipment purchased under the terms of the Federal grant for the improvement of undergraduate education.

These recitation-labs have been incorporated into the regular curriculum and are currently being offered each semester. At each registration since the first semester experiment, the sections have been consistently oversubscribed. Additional sections will be made available as faculty members of the Department of Accounting and Information Systems complete their own training seminars.

In addition, the subject matter of the course has been expanded. Students are familarized with the BASIC language which is then applied in class "live" using the "on-line classroom equipment". Four, of the fourteen single hour labs, include the development of elementary file structures, data files, program operation thereon and introductory systems design.

COST EFFECTIVENESS

The "on-line classroom" equipment cost approximately Twelve Thousand Dollars. Equipment includes:

One CRT terminal with 12" monitor and built in acoustic coupler.

One controller with terminal interface connection, 15 switches, 15 keyboard interfaces.

Fifteen keyboards with connecting cables.

Four 18" TV monitors with rolling stands and connecting cables.

One printing termina - 132 character per line capability - with connecting cables.

Approximately 850 students per semester are exposed to the computerized accounting system during the first semester accounting course. Some 90 students per semester participate actively in the weekly computer science-accounting labs. It is expected that this number will be doubled by fall 1978. The Department of English at Queens College presently offers a course in editing and uses the "on-line classroom" equipment to provide students in this course with hands-on exposure to on-line text editing.

Telephone costs are kept to a minimum.

The basic goal of all the development activity described herein is to totally integrate the computer and its use throughout the Accounting and Information System curriculum.

It appears on the basis of the first in-depth introduction to EDP in the first semester course in Accounting that the introduction of actual computer functions including time sharing on-line terminal input and output in the classroom has created an exciting and meaningful substantive addition to this level of accounting studies.

An interactive computerized management game has been installed and is being used each semester in an advanced undergraduate accounting seminar. Interactible CAI in linear programming has been added to the department's computer library. As funds become available it is planned to make use of systems such as FALCON/FORECAST and APL/PLUS available through commercial time-sharing services.

BIBLIOGRAPHY

1. Sweeney, Daniel L. (Ed.), Education for Expanding Computer Curriculums: A Symposium, American Institute of Certified Public Accountants, New York, New York, 1976.

2. Couger, J. Daniel, "Computer and the Schools of Business", Decision Line, American Institute for Decision Sciences, Sept. 1977, Vol. 8, No. 4

APPENDIX

DEBIT ROUTINES

Code Inputs for various routines available in the "Debit" Interactive Accounting System

INPUT CODE	SUBSYSTEM ROUTINE
'TRBL'	Writes trial balance to terminal
'TBTL'	Writes trial balance totals only
'ACCT'	Looks up and writes individual account balances
'WINT'	Looks up and writes perpetual inventory; all or specifically requested line item(s).
'PJ'	Prompts for and processes entries to purchase journal - at same time updates inventory and accounts payable.
'WPJ'	Writes purchase journal in detail in multicolumnar format.
'CPJ'	Prompts for and processes entries to cash payments journal - updates cash account and accounts payable.
'WCPJ'	Writes cash payments journal in detail in multicolumnar format to terminal.
'SJ'	Prompts for and processes entries to sales journal - updates sales account, accounts receivable inventory, and cost of goods sold.
'WSJ'	Writes sales journal in detail in multicolumnar format to terminal.
'BILL'	Writes statements of customers' accounts.
'CR'	Prompts for and processes entries to cash receipts journal - updates cash account, accounts receivable, and/or other accounts to be credited.
'WCR'	Writes cash receipts journal in detail in multicolumnar format.
'GJ'	Entries to general journal - maximum of three debits and three credits for each entry, routine checks for equality of debits and credits.
'BS'	Writes balance sheet.
'IS'	Writes income statement.
'STR'	Rewinds files and restores data files to original condition i.e., beginning trial balance and beginning inventory before next run.

CASH FLOW:

AN INTERACTIVE APPROACH DESIGNED TO ASSIST IN STUDENT UNDERSTANDING OF THE PRINCIPLES OF CASH MANAGEMENT

Richard T. Nyerges and Harry L. Reif

The University of Michigan-Flint

The purpose of this paper is to describe *CASH FLOW*, a program which may be used in the instruction of a Principals of Finance course. Currently, the program is being employed as part of a package of routines which form the basis for a second-level Managerial Finance course. The *CASH FLOW* program however, is flexible enough to be used on a stand-alone basis to illustrate both the construction of a cash budget and its usefulness in evaluating the impact of various financial decisions on the continued operation of the firm. It is on this latter, stand-alone context, that this paper will focus.

Our paper will be presented in two parts. First we shall discuss the application of the program to cash management decisions, and next we shall discuss, somewhat generally, a few of the technical aspects of the program. We shall forego any discussion of the issue of teaching programming vs. teaching the functional material. Throughout the planning and writing of the program, we have assumed an almost total lack of knowledge relative to all aspects of the computer. That is, we have assumed from the student's point of view, that if they are somehow able to gain the attention of the computer via a terminal and properly respond to questions prompts, they will get information which is necessary to use in completing the assignments. To put it another way, the computer is extremely useful in providing data, but it is not necessary for the student to understand the actual mechanics of how the machine produces that data.

A major topic in any Principles of Finance course is planning and control, both in terms of the general principles involved and in terms of specific areas of application. Because of the fundamental importance of cash to the firm, an excellent way to introduce the concepts of planning and control is within a discussion of cash management, specifically cash budgeting.

The topic of cash budgeting can be approached on two levels. An introductory and somewhat superficial level concentrates primarily on the construction of the cash budget, tracing the timing of cash received and expended by the firm. A more in-depth view concentrates attention on the timing of the cash flows examining not only the impact of these flows on the present and continued operations of the firm, but also on the rudimentary decisions which give rise to the flows. In most texts (and we suspect, many classroom sessions) discussion of cash budgeting is conducted

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

at the introductory level. That is, the student is guided through a physical creation of the budget and asked to note the effects of the flows on the firm's cash position and/or its borrowing requirements.

In an introductory course the time constraints of an academic calendar and the beginning student's general lack of familiarity with financial decision-making principles, this typical approach is a reasonable start. Unfortunately, the discussion rarely gets beyond this starting position before the class presses on to some other "new and exciting point". Our point here (and in classroom presentations) is that the typical approach is just half the story. The cash budget is an extremely powerful and useful tool and by extending the analysis a bit, it is possible to use this tool to its fullest extent. Thus, by delving into the roots of the flows (the "given" decisions) the student can gain insight not only into the relationships between these decisions and the magnitude and timing of the cash flows, but also into the relation between decisions and the resulting impact on the continuing operations of the firm. The major objective of the cash flow program is to afford the user the opportunity to take apart various "given decisions" one or more at a time and to note their impact in the firm. A sample run is included as Appendix A.

The program, as currently designed, operates with eleven different decision variables.[1] Several of the specified variables may be modified at the discretion of the student (user). A general schematic outline of the program appears as Figure One. The program flowchart appears as Figure Two.

As noted in Figures One and Two, the program has two basic modes of operation. These may be referred to as the *sample run mode* and the *working run mode*. Remember, the main purpose of the program is to take the student beyond a superficial treatment of merely constructing the cash budget. In concentrating on this purpose, though, we do not wish to totally ignore this aspect of the learning situation. Hence, the sample run is geared toward both illustrating the concept of the cash budget (i.e., its construction in a logical fashion) and the timing of the projected cash flows, and the impact of those flows on the potential borrowings of the firm. The real heart of the program, though, lies in the *working run mode*. Here the student/user may modify the basic decisions which give rise to the cash flow and note the impact of these modifications on the continued operations of the firm.

The student is guided into the *working run mode* by the classroom assignment accompanying the program; in this instance, the student is required to evaluate the appropriateness (in terms of optimality) of the cash budget produced in the *sample run mode*.[2]

> Part One (of the assignment) is to analyze the cash flow position of Zog Manufacturing. Specifically, you are to be concerned with whether or not Zog has an optimal cash flow. If, in your opinion, Zog's cash flow is optimal, explain why it is optimal; i.e., why it cannot or should not be changed. If the cash flow is not optimal, explain why it is not optimal and what changes Zog should undertake to produce an optimal cash flow.

Invariably, the student will pronounce the cash budget as lacking in some essential quality and thus will be off and running in an attempt to produce the most appropriate or optimal form of the budget.

---

[1] Total sales, credit sales, cash sales, purchases, capital expenses, wages, other expenses, and taxes.

[2] Use of the term optimal with respect to the process of budgeting is subject to question. (Can a budget, used as an evaluatory tool, be optimal?) It is, however, used in wording the assignment because of its thought-provoking characteristics.

FIGURE ONE

Schematic Outline:

CASH FLOW

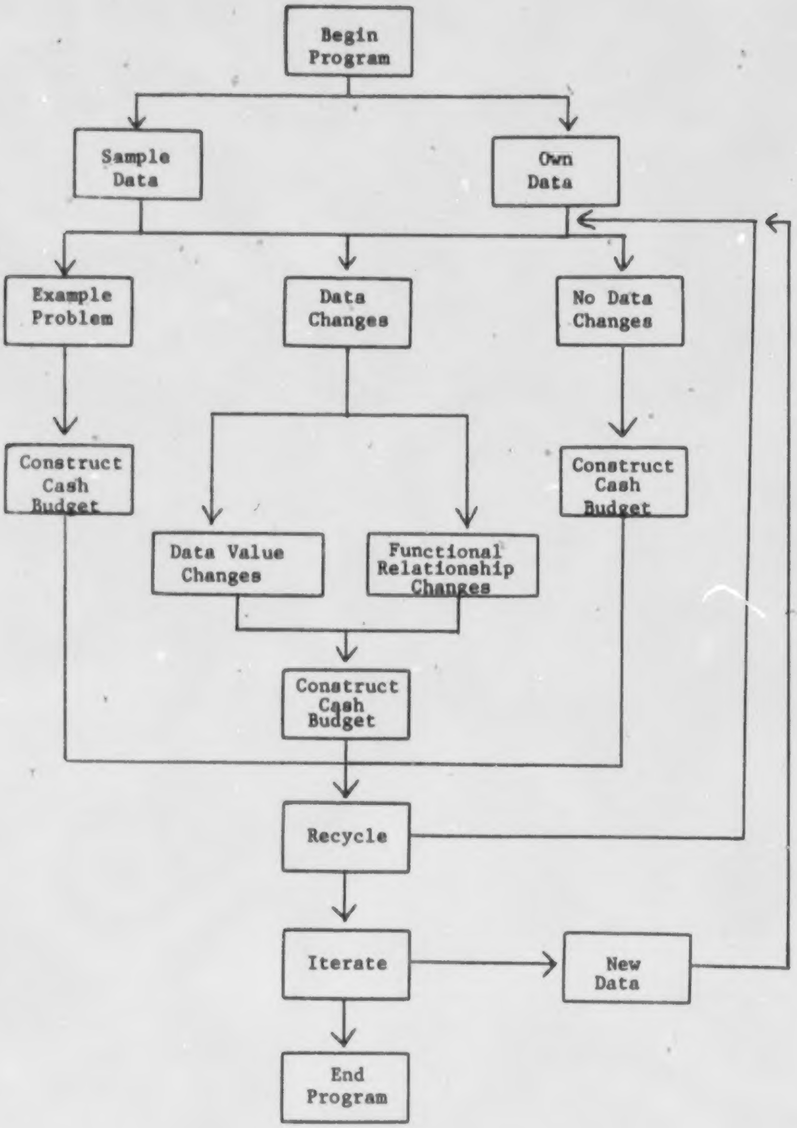
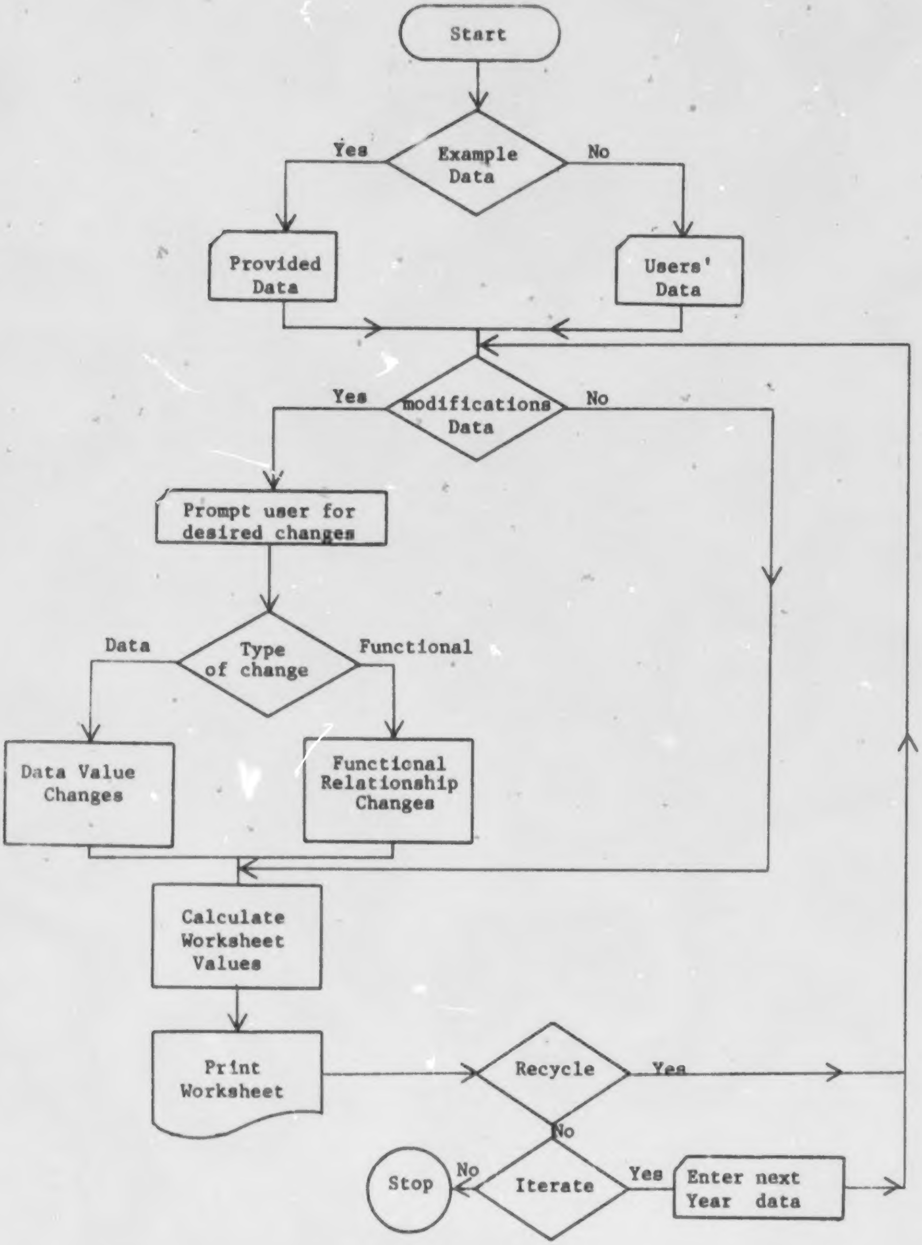

Begin Program
Sample Data
Own Data
Example Problem
Data Changes
No Data Changes
Construct Cash Budget
Data Value Changes
Functional Relationship Changes
Construct Cash Budget
Construct Cash Budget
Recycle
Iterate
New Data
End Program

FIGURE TWO
Flowchart:
CASH FLOW


Start
Example Data
Yes
No
Provided Data
Users' Data
modifications Data
Yes
No
Prompt user for desired changes
Type of change
Data
Functional
Data Value Changes
Functional Relationship Changes
Calculate Worksheet Values
Print Worksheet
Recycle
Yes
No
Iterate
Yes
No
Stop
Enter next Year data

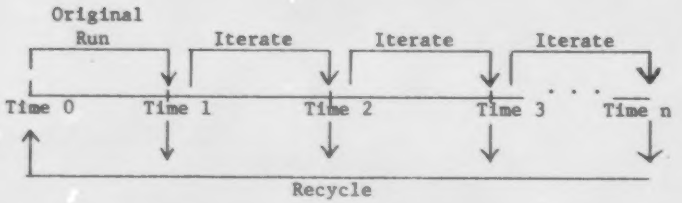
As was noted previously, there are eleven variables used in the construction of the cash budget. The student must concentrate attention on these variables to produce any changes in the cash budget. In a dual attempt to provide an initial test of student logic and to provide some degree of realism to the problem, the variables for which change is allowed are not identified. Thus, the student is forced at one extreme to logically determine which variables might reasonably be endogenous to the firm's environment, and at the other extreme (that which most frequently occurs) to experiement to determine the "rules of the game". At this point, the student begins to gain some insight into the relationships which exist between the decisions which the financial manager makes and the resulting impact of these decisions upon the operations of the firm.

As an example, take the category of sales. Quite often students determine that the cash inflows are too low and that the way to remedy the situation is to increase sales. Following their logic, they usually request a modification (an increase) in sales. What a wonderfully simplistic solution: revenues are not great enough, and, all that must be done is increase sales. Their real world counterpart (in this case the computer) responds by noting that one does not ordinarily wave a magic wand and increase sales. It does inform the user, however, that price is an acceptable change variable. Predictably, the following response is to increase price. Often the student is surprised to find that revenues fall with large price increases. By logic, deduction, or again, more frequently by experimentation, the student is forced into an analysis of demand for the product. The variable "sales" is thus designed to illustrate both the exogenous demand relation (i.e., given a *ceteris paribus* situation, the relationship between the price of the product and the quantity demanded) and the impact of pricing decisions on the overall level of activity and, on the cash position of the firm.

Time is an extremely important parameter and may be introduced into the model by requesting a multiple-period run (i.e., by requesting an iteration). This program branch allows the students to carry their projects beyond a single forecasting cycle (in this case, one year). By so doing, the student can trace the impact of a single or a series of modifying decisions through time. Should a modification prove less than satisfactory, the student may "erase" the decision by recycling back to time period zero. Note that it is not possible to erase just one time period. See Figure Three.

FIGURE THREE

Iteration and Recycling



In its most flexible form, the program allows introduction of the original data base and modification of the specified functional relationship of the variables. Though the model is reasonably powerful in this branch, it is generally too complex to be particularly useful to an introductory class.

CASH FLOW is a modular interactive FORTRAN program designed to be as system independent as possible. The current version of the program is running on an Amdahl 470V/6. Two extensions of IBM FORTRAN have been used, both connected with program I/O. The PRINT statement and the END= option on the READ statement have been used and could easily be programmed out if the user's particular installation does not support these extensions. The modules consist of a run deck, a data file (which can be altered by the instructor), and a scratch file to be used by students for inputting their *own* data. *CASH FLOW* is written as a series of subroutines for ease of modification should the user wish to change the program to better suit his needs or should it become necessary to omit certain options in the program due to restrictions inherent in the user's computer.

Also included in *CASH FLOW* are many "hooks" which were inserted to provide for future expansions and modifications to the program. A programmer's guide for *CASH FLOW* is not yet developed. For the present time the program has been adequately commented to allow for programmer modifications if desired.

While meaningful execution of the program is generally dependent upon the student having read an introductory unit beforehand, it is possible for a novice to successfully execute the program due to the many input validation and error-checking capabilities of the program. As the user becomes more experienced, he will find that the program grows more and more flexible. Through the use of default settings and values, we have given the user the option of executing a simulation of an entire cash flow analysis simply by answering yes/no to a number of prompts. The same program will let the more sophisticated user perform iterations on the data as well as modify some of the initial data values. For the user who would like to use his own data, we have also allowed that option, provided that the data is entered according to a specified format.

Data provided with the program consists of some initial dollar values along with values that denote the relationship to other data values as well as lag or lead times when applicable. The data file is designed so as to be easily modified by the instructor.

CASH FLOW is the first in a set of tools that will eventually be integrated to form a financial analysis package. This package will find applications in both the classroom and the real world. The package is designed to be a flexible and integral part of the Principles of Finance classroom environment, as well as a tool for business managers to aid in predicting the outcomes of financial decisions. As such, it will be a thinking tool for the student and a working tool for the financial manager.

APPENDIX A

Example One presents a typical student run of *CASH FLOW*. In our setting, the student has been prepared for this run in the classroom discussions on cash budgeting and by reading the unit which accompanies the program. We do not feel, however, that this level of preparation is necessary to successfully use the program. In our experience, though, the preparation greatly enhances the learning process. In Example One the lowercase type denotes user responses while the uppercase type indicated computer replies or instructions.

EXAMPLE ONE
Sample Student Run

```
 ARE YOU GOING TO USE THE SAMPLE DATA? (YES/NO)
yes

DO YOU WANT TO MODIFY ANY OF THE DATA? (YES/NO)
no


WORKSHEET:

                        JAN      FEB      MAR      APR      MAY      JUN
TOTAL SALES             800.     500.     600.     700.     900.    1200.
CASH SALES              160.     100.     120.     140.     180.     240.
CREDIT SALES            936.     576.     360.     432.     504.     648.
CREDIT SALES2            80.     104.      64.      40.      48.      56.
TOTAL RECEIPTS         1176.     780.     544.     612.     732.     944.

PURCHASES               640.     400.     480.     560.     720.     960.
CASH PUR/PAY            160.     100.     120.     140.     180.     240.
CRED PUR/PAY            780.     480.     300.     360.     420.     540.
WAGES                    90.      75.      80.      85.      95.     110.
TAX                       0.       0.     250.       0.       0.     250.
OTHER EXP                50.      50.      50.      50.      50.      50.
CAPITAL EXP.              0.       0.       0.     500.       0.       0.
TOTAL DISBURSE.        1080.     705.     800.    1135.     745.    1190.

CASH BUDGET:

NET FLOWS                96.      75.    -256.    -523.     -13.    -246.
CASH BOM                150.     246.     321.     100.     100.     100.
NET CASH                246.     321.      65.    -423.      87.    -146.
REQUIRED                100.     100.     100.     100.     100.     100.
BORROWING                 0.       0.      35.     523.      13.     246.
CUMUL BOR.                0.       0.      35.     558.     571.     817.
CASH EOM                246.     321.     100.     100.     100.     100.
-------------------------------------------------------------------------


WORKSHEET:

                        JUL      AUG      SEP      OCT      NOV      DEC
TOTAL SALES            1300.     900.    1000.    1000.    1200.    1500.
CASH SALES              260.     180.     200.     200.     240.     300.
CREDIT SALES            864.     936.     648.     720.     720.     864.
CREDIT SALES2            72.      96.     104.      72.      80.      80.
TOTAL RECEIPTS         1196.    1212.     952.     992.    1040.    1244.
```

```
PURCHASES               1040.   720.   800.   800.   960.  1200.
CASH PUR/PAY             260.   180.   200.   200.   240.   300.
CRED PUR/PAY             720.   780.   540.   600.   600.   720.
WAGES                    115.    95.   100.   100.   110.   125.
TAX                        0.     0.   250.     0.     0.   250.
OTHER EXP                 50.    50.    50.    50.    50.    50.
CAPITAL EXP.               0.   500.     0.     0.     0.     0.
TOTAL DISBURSE.         1145.  1605.  1140.   950.  1000.  1445.


CASH BUDGET:

NET FLOWS                 51.  -393.  -188.    42.    40.  -201.
CASH BOM                 100.   100.   100.   100.   100.   100.
NET CASH                 151.  -293.   -88.   142.   140.  -101.
REQUIRED                 100.   100.   100.   100.   100.   100.
BORROWING                -51.   393.   188.   -42.   -40.   201.
CUMUL BOR.               766.  1159.  1347.  1305.  1265.  1466.
CASH EOM                 100.   100.   100.   100.   100.   100.

------------------------------------------------------------

DO YOU WANT TO RECYCLE? (YES/NO)
no

DO YOU WANT TO ITERATE? (YES/NO)
no
↑rerun
↑EXECUTION BEGINS

 ARE YOU GOING TO USE THE SAMPLE DATA? (YES/NO)
yes

DO YOU WANT TO MODIFY ANY OF THE DATA? (YES/NO)
yes

WHICH TYPE OF MODIFICATION DO YOU WANT TO MAKE?
IF YOU ARE NOT SURE MERELY PUSH 'RETURN'.


DO YOU WISH TO MODIFY ANY OF THE INITIAL DATA? (YES/NO)
yes

HOW MANY VARIABLES DO YOU WISH TO MODIFY?
(AT LEAST 1 BUT NOT MORE THAN 8)
2

SPECIFY THE VARIABLE(S), ONE AT A TIME,
THAT YOU WISH TO MODIFY AND INDICATE,
WHEN PROMPTED, HOW YOU WISH TO MODIFY THEM.
THEY MUST COME FROM THE FOLLOWING LIST: SALE, CRED,
CASH, PURC, CAPI, WAGE, OTHE, TAX
? sale

YOU CANNOT DIRECTLY CHANGE SALES, HOWEVER
YOU MAY INCREASE OR DECREASE PRICE.  DO YOU WISH TO
DO SO? (YES/NO)
yes
PICK ANOTHER VARIABLE. ? tax

YOU ARE REQUIRED TO PAY UNCLE SAM,
IF YOU TRY TO ALTER THIS PAYMENT, NASTY THINGS
WILL OCCUR IN LATER PERIODS.
```

```
DO YOU WANT TO CHANGE THE PRICE? (INCREASE/DECREASE/NO)
increase

BY WHAT PERCENTAGE?
 33

WORKSHEET:

                        JAN     FEB     MAR     APR     MAY     JUN
TOTAL SALES             355.    222.    266.    310.    399.    532.
CASH SALES               71.     44.     53.     62.     80.    106.
CREDIT SALES            936.    255.    160.    192.    223.    287.
CREDIT SALES2            80.    104.     28.     18.     21.     25.
TOTAL RECEIPTS         1087.    404.    241.    271.    325.    419.


PURCHASES               284.    177.    213.    248.    319.    426.
CASH PUR/PAY             71.     44.     53.     62.     80.    106.
CRED PUR/PAY            780.    213.    133.    160.    186.    239.
WAGES                    68.     61.     63.     66.     70.     77.
TAX                       0.      0.    250.      0.      0.    250.
OTHER EXP                50.     50.     50.     50.     50.     50.
CAPITAL EXP.              0.      0.      0.    500.      0.      0.
TOTAL DISBURSE.         969.    368.    549.    837.    386.    722.

CASH BUDGET:

NET FLOWS               118.     35.   -308.   -566.    -61.   -304.
CASH BOM                150.    268.    304.    100.    100.    100.
NET CASH                268.    304.     -5.   -466.     39.   -204.
REQUIRED                100.    100.    100.    100.    100.    100.
BORROWING                 0.      0.    105.    566.     61.    304.
CUMUL BOR.                0.      0.    105.    670.    732.   1036.
CASH EOM                268.    304.    100.    100.    100.    100.
--------------------------------------------------------------------


WORKSHEET:

                        JUL     AUG     SEP     OCT     NOV     DEC
TOTAL SALES             576.    399.    443.    443.    532.    665.
CASH SALES              115.     80.     89.     89.    106.    133.
CREDIT SALES            383.    415.    287.    319.    319.    383.
CREDIT SALES2            32.     43.     46.     32.     35.     35.
TOTAL RECEIPTS          530.    537.    422.    440.    461.    552.


PURCHASES               461.    319.    355.    355.    426.    532.
CASH PUR/PAY            115.     80.     89.     89.    106.    133.
CRED PUR/PAY            319.    346.    239.    266.    266.    319.
WAGES                    79.     70.     72.     72.     77.     83.
TAX                       0.      0.    250.      0.      0.    250.
OTHER EXP                50.     50.     50.     50.     50.     50.
CAPITAL EXP.              0.    500.      0.      0.      0.      0.
TOTAL DISBURSE.         563.   1046.    700.    477.    499.    835.
```

```
CASH BUDGET:

NET FLOWS                -33.   -508.   -278.    -37.    -38.   -284.
CASH BOM                 100.    100.    100.    100.    100.    100.
NET CASH                  67.   -408.   -178.     63.     62.   -184.
REQUIRED                 100.    100.    100.    100.    100.    100.
BORROWING                 33.    508.    278.     37.     38.    284.
CUMUL BOR.              1069.   1577.   1855.   1892.   1930.   2214.
CASH EOM                 100.    100.    100.    100.    100.    100.

---------------------------------------------------------------------

DO YOU WANT TO RECYCLE? (YES/NO)
no

DO YOU WANT TO ITERATE? (YES/NO)
yes

ENTER 12 NEW VALUES, WITH DECIMAL POINTS, SEPARATED BY COMMAS FOR
TOTAL SALES
ENTER 'NO' FOR NOCHANGE.
no

ENTER 12 NEW VALUES, WITH DECIMAL POINTS, SEPARATED BY COMMAS FOR
TAX
ENTER 'NO' FOR NOCHANGE.
10.,10.,25.,10.,10.,25.,10.,10.,25.,10.,10.,25.

ENTER 12 NEW VALUES, WITH DECIMAL POINTS, SEPARATED BY COMMAS FOR
OTHER EXP
ENTER 'NO' FOR NOCHANGE.
no

ENTER 12 NEW VALUES, WITH DECIMAL POINTS, SEPARATED BY COMMAS FOR
CAPITAL EXP.
ENTER 'NO' FOR NOCHANGE.
no

ENTER 12 NEW VALUES, WITH DECIMAL POINTS, SEPARATED BY COMMAS FOR
REQUIRED
ENTER 'NO' FOR NOCHANGE.
no

DO YOU WANT TO MODIFY ANY OF THE DATA? (YES/NO)
no


WORKSHEET:

                         JAN     FEB     MAR     APR     MAY     JUN
TOTAL SALES              355.    222.    266.    310.    399.    532.
CASH SALES                71.     44.     53.     62.     80.    106.
CREDIT SALES             479.    255.    160.    192.    223.    287.
CREDIT SALES2             43.     53.     28.     18.     21.     25.
TOTAL RECEIPTS           592.    353.    241.    271.    325.    419.
```

```
PURCHASES              284.   177.   213.   248.   319.   426.
CASH PUR/PAY            71.    44.    53.    62.    80.   106.
CRED PUR/PAY           399.   213.   133.   160.   186.   239.
WAGES                   68.    61.    63.    66.    70.    77.
TAX                     10.    10.    25.    10.    10.    25.
OTHER EXP               50.    50.    50.    50.    50.    50.
CAPITAL EXP.             0.     0.     0.   500.     0.     0.
TOTAL DISBURSE.        598.   378.   324.   847.   396.   497.


CASH BUDGET:

NET FLOWS               -5.   -25.   -83.  -576.   -71.   -79.
CASH BOM               100.   100.   100.   100.   100.   100.
NET CASH                95.    75.    17.  -476.    29.    21.
REQUIRED               100.   100.   100.   100.   100.   100.
BORROWING                5.    25.    83.   576.    71.    79.
CUMUL BOR.            2220.  2245.  2328.  2904.  2975.  3054.
CASH EOM               100.   100.   100.   100.   100.   100.
--------------------------------------------------------------


WORKSHEET:

                       JUL    AUG    SEP    OCT    NOV    DEC
TOTAL SALES            576.   399.   443.   443.   532.   665.
CASH SALES             115.    80.    89.    89.   106.   133.
CREDIT SALES           383.   415.   287.   319.   319.   383.
CREDIT SALES2           32.    43.    46.    32.    35.    35.
TOTAL RECEIPTS         530.   537.   422.   440.   461.   552.


PURCHASES              461.   319.   355.   355.   426.   532.
CASH PUR/PAY           115.    80.    89.    89.   106.   133.
CRED PUR/PAY           319.   346.   239.   266.   266.   319.
WAGES                   79.    70.    72.    72.    77.    83.
TAX                     10.    10.    25.    10.    10.    25.
OTHER EXP               50.    50.    50.    50.    50.    50.
CAPITAL EXP.             0.   500.     0.     0.     0.     0.
TOTAL DISBURSE.        573.  1056.   475.   487. . 509.   610.

CASH BUDGET:

NET FLOWS              -43.  -518.   -53.   -47.   -48.   -59.
CASH BOM               100.   100.   100.   100.   100.   100.
NET CASH                57.  -418.    47.    53.    52.    41.
REQUIRED               100.   100.   100.   100.   100.   100.
BORROWING               43.   518.    53.    47.    48.    59.
CUMUL BOR.            3097.  3616.  3669.  3716.  3764.  3823.
CASH EOM               100.   100.   100.   100.   100.   100.

--------------------------------------------------------------

DO YOU WANT TO RECYCLE? (YES/NO)
no

DO YOU WANT TO ITERATE? (YES/NO)
no
↑EXECUTION TERMINATED
↑
```

MUSICAL ACOUSTICS EXPERIMENTS WITH A LABORATORY-BASED COMPUTER

G. T. Wood

Cleveland State University

Introduction

A laboratory program is an especially important part of an undergraduate musical acoustics course since this is where the concepts and special techniques presented in the text can be demonstrated. In building up a set of laboratory experiments some can be developed from the equipment already available in the elementary physics laboratory such as springs, string vibrators, microphones, loudspeakers, amplifiers, oscillators, oscilloscopes and air columns. However, some techniques which are emphasized in musical acoustic texts such as the measurement of reverberation time, harmonic analysis, or the measurement of resonance curves require the use of sophisticated equipment if useful results are to be obtained. Providing multiple setups of such specialized equipment for an hour laboratory of 32 students such as we have at Cleveland State University may not be justified. However, once an on-line computer system is established in the laboratory, multiple setups of these instruments can be made available to the students using in most cases only general purpose equipment such as amplifiers, microphones, and loudspeakers. In effect some of the specialized functions of the equipment are achieved by computer programing.

Computer Hardware

A computer on line to elementary physics laboratories has been under development at Cleveland State University for several years. The basic computer installation consists of a Texas Instrument TI 980A Computer (64K 16-Bit memory) with a moving-head disk, card reader, line printer, magnetic tape drives, card punch and a CalComp Plotter. The TI 980A is coupled to eight stations in a laboratory of 32-student capacity. Each station is supplied with a keyboard and a TV monitor on which 16 lines of 32 characters may be displayed. Information may also be output independently at each station on 40 character/sec hard-copy printers. In addition time and voltage sensors make possible on-line measurements. Each station provides

1. Two inputs to record event times
2. One input to record voltage samples from a waveform or transient.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

Graphical information may be output to the TV monitor on a 64 by 48 point raster. In addition two voltage outputs per station are provided for generating wave forms or for producing graphical displays on an oscilloscope. Both the TV display and the voltage outputs are refreshed cyclically from a core memory accessible with standard computer instructions. However, memory cycles associated with character or voltage refresh cycles are independent from computer memory cycles and do not steal computing time.

Computer Software

Our on-line laboratories are run in FORTRAN backed up with assembly language programs including special I/O subroutines linked with the main program and the TI BASIC OPERATING SYSTEM. This is a single user system which we have adapted in such a way that the 8 stations independently progress through the same program, thereby, maximizing efficiency in the use of memory space. At the beginning of the FORTRAN Main program the operation is single user mode until a subprogram called BRANCH is called. BRANCH creates separate user memory spaces for each station for storing status, return addresses, keyboard buffers, pointers, etc. Also some instructions in the subroutines produced by the FORTRAN compiler (such as Global Setup and the Floating Point Processor) are overlaid in order that subroutine returns for multiple users are handled and Run time FORTRAN errors such as divide check and input format errors on the keyboards are taken care of without causing a return to the operating system. After returning from BRANCH the main program is executed for Station 1 until a keyboard, event timer or voltage input action is pending. At this time "round robin" polling occurs and a switch is made to the next station in line.

Among the various subroutines written in assembly language and callable from the FORTRAN program four are briefly discussed below since they are required for musical acoustic experiments involving voltage input to or output from the computer.

1. TIMER (TIME, NSTAT, NTIMES)
2. ADC (NPTS, BUFF, NRMS)
3. DAC (IX, IY)
4. DACINT (INT, NPTS)

TIMER sets the interupt system to receive NTIMES time events on station NSTAT and stores the time of each event received in an array TIME. ADC sets the interupt system to make NPTS measurements of voltage into the data buffer BUFF. If NRMS is greater than zero each measurement will be the root-mean-square (rms) average of NRMS successive samples. If NRMS = 0 no averaging is performed. The interupts are in synchronization with the voltage output samples as initialized in DACINT below. DAC transfers the data in the integer arrays IX and IY to the voltage output refresh memory. DACINT initializes the voltage refresh memory. INT is the time between voltage output samples (in units of 0.25 microseconds) and NPTS is the number of samples in the periodic cycle of output samples.

Reverberation Time Experiment

Reverberation Time is an important parameter for characterizing an auditorium or room used for music or speech purposes. It is defined as the time required for the intensity of sound in a room to fall a factor of one million following a loud short burst of sound. In performing this experiment the student reads instructions on the TV monitor. The final message reads:

CHOOSE A MEMBER OF YOUR GROUP TO
PRODUCE THE SOUND BURST. HE WILL
MAKE A LOUD SOUND ABOUT A FOOT
AWAY FROM THE TRIGGER MICRO-
PHONE. JUST BEFORE HE IS READY
TO MAKE THE SOUND PUSH THE KEYS
GO ON YOUR KEYBOARD. YOU

```
SHOULD NOT TRY TO PICK UP THE
SOUND BURST PRODUCED BY ANOTHER
GROUP SINCE ONLY ONE STATION
CAN RECORD SOUNDS AT A TIME.
ROOT-MEAN-SQUARE VALUES OF YOUR
SOUND WILL BE PRINTED ON THE TV
AND GRAPHED ON THE SCOPE AT 0.01
SECOND INTERVALS.
```

After a student keys in the letters "GO" the program advances to the FORTRAN Call statements

```
CALL TIMER (TIME, 1, 1)
CALL ADC (NPTS, BUFF, NRMS)
```

When a student has produced a sound by clapping two wooden boards together a signal from a trigger microphone placed close to the sound source is input to the event timer No. 1. Since the TIMER subroutine is set up to receive one event on Timer input No. 1 the program then advances to the ADC subroutine which makes NPTS root-mean-square voltage measurements. Each of these measurements is a root-mean-square average of NRMS successive voltage samples collected from the microphone connected to the active station. The voltage samples are collected on an interupt basis (every 100 microseconds in this experiment as set by an earlier call to DACINT). Before plotting the root-mean-square measurements versus time the students estimate the background noise by averaging the last 10 measurements and subtracting this background value from each of the earlier measurements. A typical corrected plot is shown in Fig. 1 on a semi-log plot. The rms voltage is proportional to the rms pressure fluctuation at the microphone which in turn is proportional to the square-root of sound intensity. Therefore, the reverberation time is obtained from the time for 3 decades of decay of the background-corrected voltage measurements. For the data in Fig. 1 this corresponds to a reverberation time of 0.53 sec in good agreement with the value of 0.61 sec estimated by the student for the laboratory room from the Sabin Formula.[1] The Sabin estimate is based on the dimensions of the room and the handbook values of the absorption coefficients of the ceiling, wall and floor.

Harmonic Sound Analysis Experiment

In this experiment the student measures the harmonic spectrum of a periodic complex tone produced by a musical instrument or the voice. Each student is asked to choose his own project. For example, he may decide to study the effect of the point of bow contact on the harmonic content of bowed violin sounds or he might choose to study the formant structure of different vowel sounds produced by the voice. The present system represents an evolution at CSU which began several years ago when the Musical Acoustic students took polaroid photos of oscilloscope traces, measured samples by hand, punched cards and submitted the card deck to the campus computer. Now waveform data is collected on-line and is stored in the laboratory computer memory. The data is then transferred to the voltage-output refresh memory. At each station the voltage outputs are connected to the X-Y inputs of an oscilloscope producing an immediate graphical display of the wave-form. Through the keyboard the students can control the position of two bright cursor dots and in this way select one complete cycle of points from the periodic waveform for analysis. When this is done the student may request the harmonic analysis of the waveform data for the selected range of points. The method involves the standard Fourier Series Analysis of a discrete data Spectrum. Although floating point operations are done with programs rather than hardware in our computer the typical analysis requires only 5 to 15 seconds of time. Since polling is occuring during the Fourier Analysis calculation, students at other stations are unaware of the heavy computer involvement in number crunching since they can continue to record wave forms or position cursors. An example of a harmonic spectrum of the vowel "e" sound is shown in Fig. 2. It was produced by a female voice at 247 Hertz.

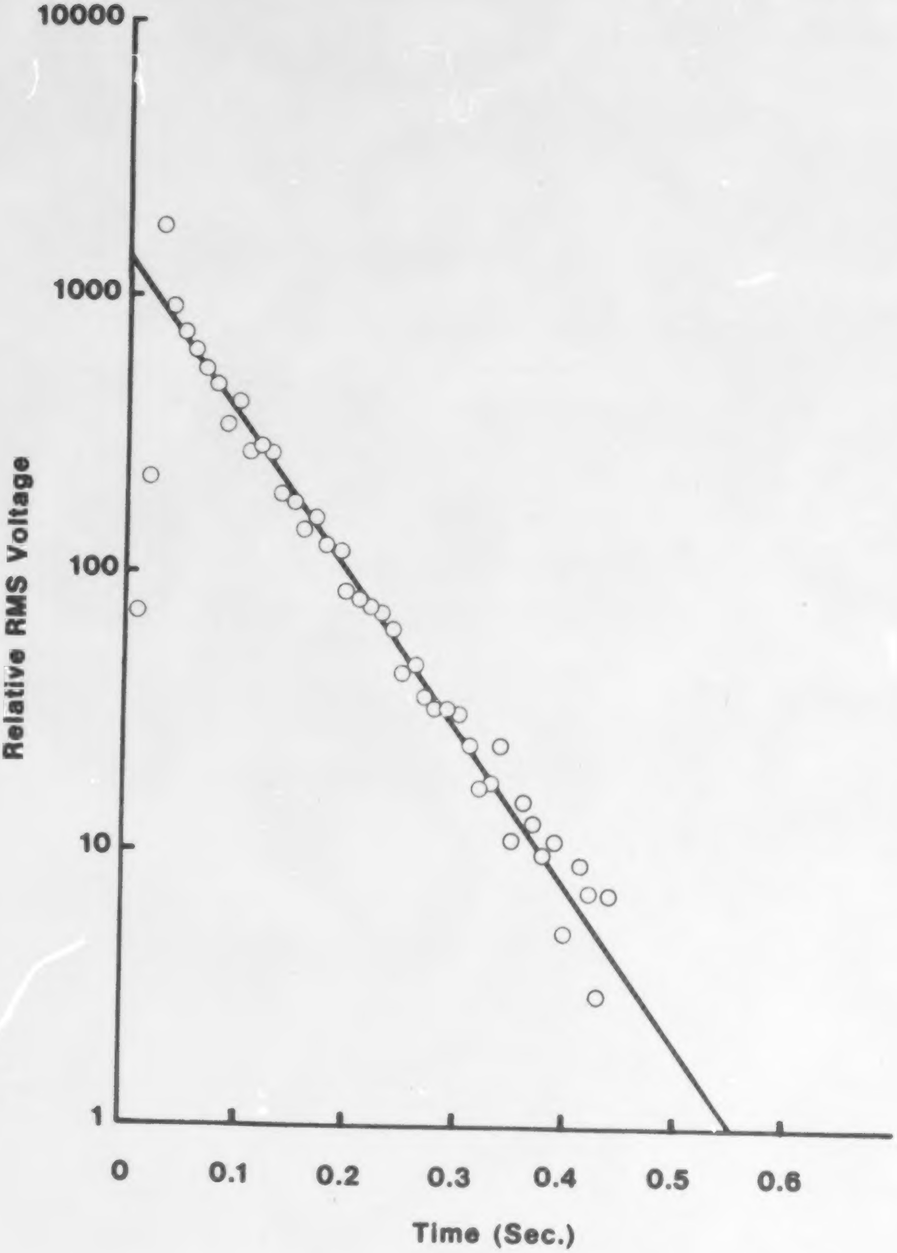


FIGURE 1. A plot of root-mean-square voltage versus time following a sound burst in a physics laboratory at Cleveland State Univ. A reverberation time of 0.53 sec was obtained from the time for a factor of 1000 decay in the rms voltage.

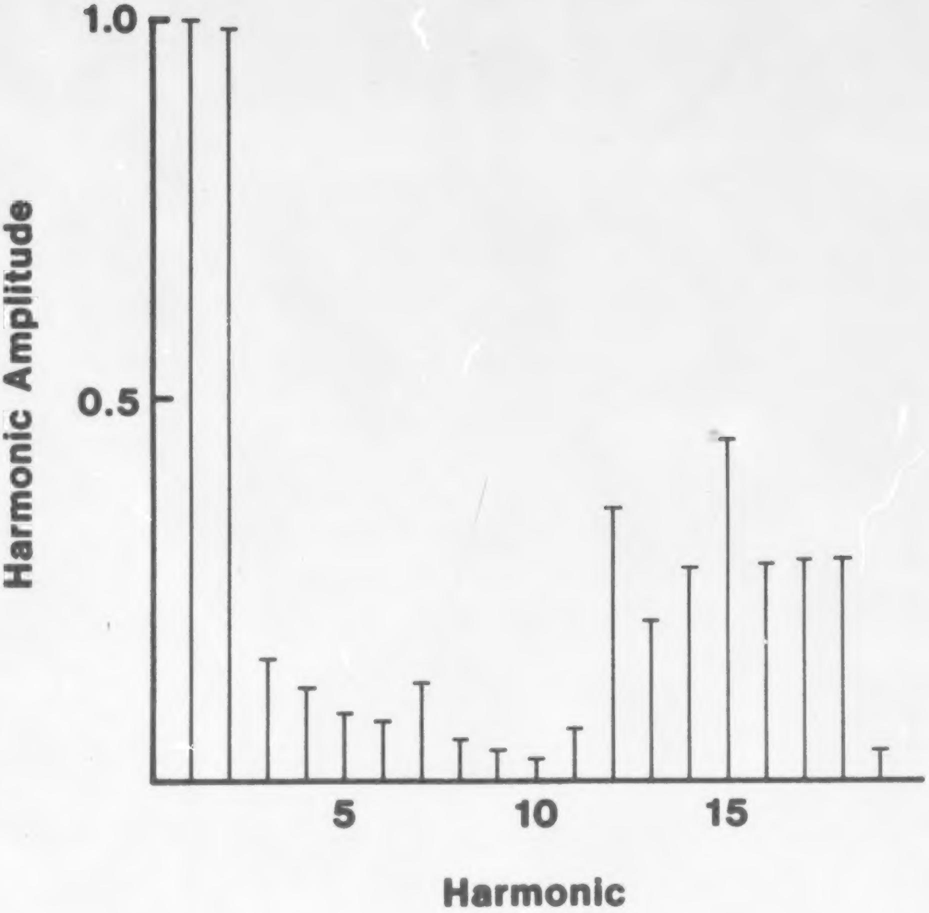


FIGURE 2. A plot of the harmonic spectrum of a vowel "e" sound produced by a student at 247 Hertz.

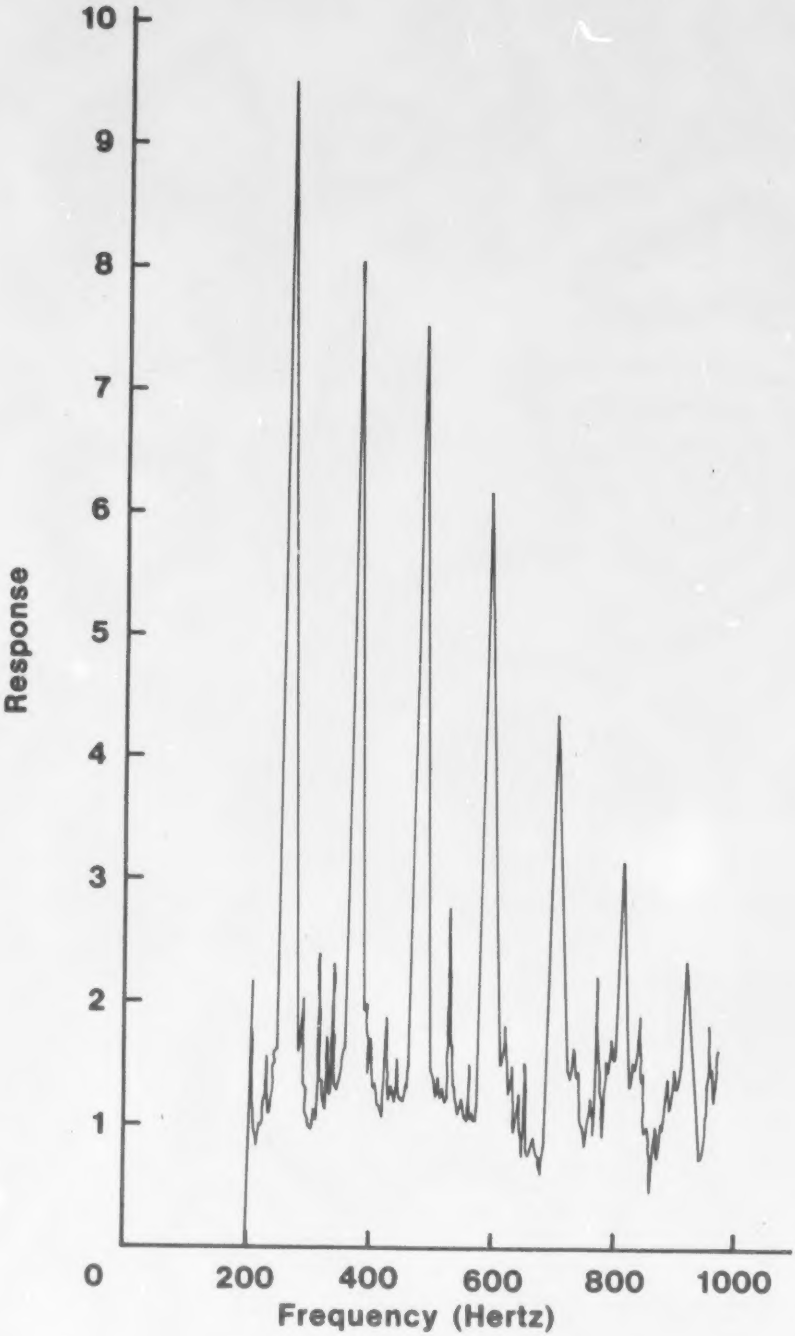


FIGURE 3. A resonance curve for a cylindrical tube. This figure is a tracing of a CalComp plot with some secondary resonances smoothed over.

The collection of voltage samples is again done with a FORTRAN Call CALL ADC (NPTS, BUFF, 0). In this case no root-mean-square averaging is performed since the third argument in the argument list of the ADC subroutine call is zero. In this experiment voltage samples were received every 60 microseconds. This is about the lower limit in sampling speed due to the time consumed by the program steps involved with the receipt of each sample.

Resonance Curve Demonstration

This program was developed recently as a demonstration and will be substituted for the present air column experiment next year. In the past the students have searched for the frequencies of air column resonances by sweeping the frequency of an audio oscillator manually. Sound waves are set up in the air column by a loudspeaker driven by the oscillator and the response from a microphone attached to the air column is viewed on an oscilloscope. In the computer version a sine waveform is produced on the voltage output of the computer. This waveform is amplified and drives the loudspeaker. The microphone is connected to the voltage input and is sampled by the computer at the same rate that voltage samples are output from the computer. With the help of the keyboard-TV terminal the student keys in the range of frequencies which are to be scanned, the frequency interval (in cents) used in each step and the time that is to be spent at each frequency. As the computer sweeps through the frequency range a root-mean-square average of the microphone response is measured (similar to the reverberation time experiment) at each frequency and plotted on the CalComp Plotter. An example of such a plot is shown in Fig. 3 for a cylindrical 7/8-in diameter copper tubing 153-cm long. A horn speaker driver incorporating a small ceramic microphone [2] was attached to one end of the tube leaving the other end open. As can be seen the main resonances are at odd harmonic frequencies as expected for an air column open at one end and closed at the other. However, the frequency range used in this particular measurement excluded the lowest two resonant frequencies. With suitable attachments the horn driver can also be used to drive various instruments such as a trumpet, clarinet or oboe. Also a loudspeaker voice coil[3] may be attached to a violin bridge while a microphone in front of the violin picks up the sound. Thus the computer drives the violin through the various wood and air resonances and a resonance curve is measured. When the computer system is used to replace the present air column experiment a bottle neck will occur since there is only one plotter to draw the resonance curves. This problem will be alleviated by splitting the process into two steps. In the first step a resonance curve is scanned and the results are stored in the computer memory. Several trials each requiring only a few seconds may be necessary before the appropriate conditions (such as amplifier gain) are achieved. In the second step the final results can be plotted on the CalComp Plotter from computer memory without stopping the work of other students doing adjustment scans.

Conclusions

The use of a computer with on-line voltage input and output makes possible the development of sophisticated musical acoustic experiments in multiple setups for large laboratories. In most cases only standard equipment such as microphones and amplifiers are needed in conjunction with the computer. A single FORTRAN program is shared by eight user stations thereby saving on memory space. Since FORTRAN compilers are available on most computers the laboratory programs can be exported. The specific hardware used for timing, voltage sensing and output need not be the same at another installation. The subroutines such as BRANCH, TIMER, ADC, and DAC together with the associated system software would have to be written in the assembly language of the computer to which the programs were being exported. Once this specialized hardware-dependent software is written the implementation of new experiments is relatively easy since this is done in FORTRAN. The use of the polling feature initialized by the program BRANCH makes it possible to operate single-user FORTRAN in a mode whereby all the student groups progress independently through the same program. The software system described above would be less satisfactory if each group wanted to do a different experiment. If this were a requirement it would be better to use a multi-user operating system although considerably more memory space would be used.

Acknowledgment

We wish to thank the National Science Foundation for support from a COSIP grant in the initial phases of this work and the Center for Effective Learning at Cleveland State University for grant support.

References

1. J. Backus, "the Acoustical Foundations of Music", W. W. Norton & Co., New York, 1977.
2. The author is indebted to A. H. Benade at Case Western Reserve University for the use of this transducer.
3. C. M. Hutchins, "Instrumentation and Methods for Violin Testing", JAES 21 (1973) 563.

COMPUTERIZED SOLAR HEATING ANALYSIS

Richard A. Blade and Michael H. Scott

University of Colorado - Colorado Springs

Introduction

In teaching solar energy heating system design and cost analysis, it is extremely useful to compare the design and cost requirements for students' own homes. This can readily be done in a most dramatic manner using a computer and inputting only the average (or peak) monthly heating bill and the kind of heating (natural gas, electricity, or LPG). Given that information from each student, the authors have developed a program for their HP9820 computer that, for that individual's home, plots the collector area; the cost of the complete solar heating system for 75 percent, 85 percent, and 95 percent of annual heating; the projected future monthly bills for his/her present heating system; and the future monthly costs assuming a bank loan is made to pay for the solar heating system. Figure 1 shows a typical plot. Reasonable assumptions are made for all the design and cost factors so no additional data need be entered, but the program readily permits the change of these assumptions so each individual student can study his/her individual situation in greater detail.

The Solar Heating System

Often solar energy texts quote the requirements for a solar heating system in terms of the floor area or the volume of heated area. This highly inaccurate because unjustified assumptions are made about the insulation, heat conservation measures such as storm windows and doors, the nature of fire places, the orientation and exposure of the house, and the like. In fact, the single best indicator for designing a retrofitted solar heating system is the heat it presently uses, as calculated from the heating bills. Admittedly, it does not take into account the fact that before retrofitting a nonelectrically-heated house, it is usually prudent to add more insulation and other heat conservation measures. It does take into account the heating habits (i.e., the setting of the thermostat) of the homeowner, but it does not reflect that a solar heating system becomes much more economical if the homeowner is willing to tolerate a ten-degree temperature variation over a 24-hour period with the lower temperature in the early morning.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

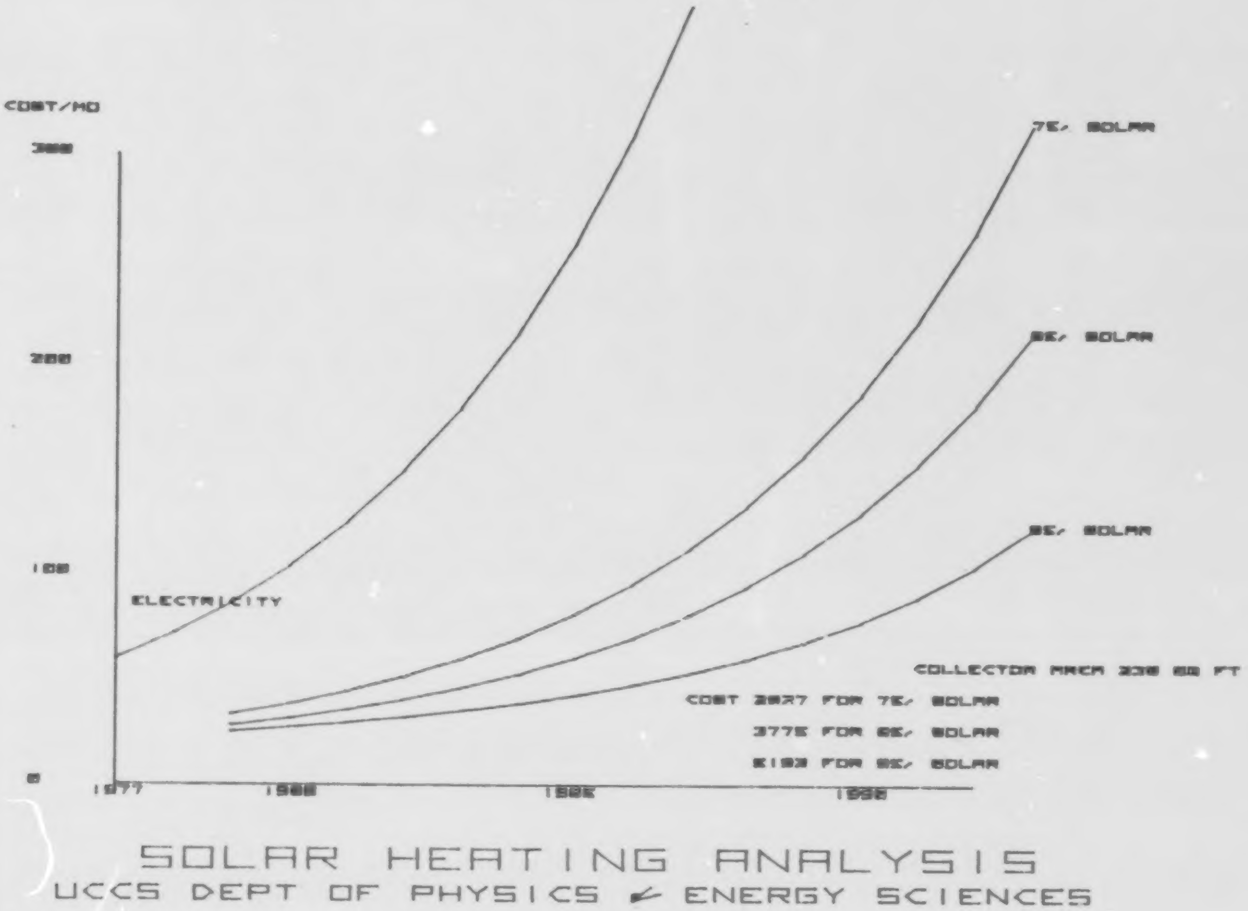


Fig. 1: Typical computer plot.

The initial step in the computer program is to calculate the average daily consumption of heating fuel using the heating bill and the kind of heat. We use the following rates[1]:

NATURAL GAS: $5.38 for the first 4000 cu. ft.
$0.109/cu. ft. after that

ELECTRICITY: $7.55 for the first 271 KWH
$0.022/KWH after that

LPG: $0.49/gal.

Then the amount of fuel is converted to Btu's of heat produced using

1000 Btu's/cu. ft. natural gas[2]
3412 Btu's/KWH electricity[3]
90,000 Btu's/gal. LPG[4]

Then, using information about our local climate, and assuming an optimal orientation of roughly southern facing collectors at 60° slope, the required collector area is calculated using 1200 Btu/ft^2 per day[5]. Clearly the exact orientation and efficiency of the collectors and the design of the system for 75 percent, 85 percent, or 95 percent solar energy make some difference, but the effect of such factors is completely overshadowed by the effect of the amount of heat storage. For this we assume water, and for this climate (the primary factor being the annual average number of consecutive-day periods where the weather does not permit significant solar collection) we assume a 75 percent solar heating system requires one-day storage, an 85 percent system requires two-day storage, and a 95 percent system requires three-day storage. Assuming the storage temperature averages 70°F above the house temperature, this translates into[6]

0.00188 gal/Btu for the 75% system
0.00469 gal/Btu for the 85% system
0.00938 gal/Btu for the 95% system

The Cost Analysis

The projected monthly heating bills for the future are readily calculated using an annual rate of increase, r, of the cost of the fuel. We have chosen

r = 35% for natural gas
= 20% for electricity
= 25% for LPG

These values are highly contraversial, and at best are short term. Anyone thinking they are excessively high, however, should look at the increases in their own heating bills over the last five years! Government regulation will affect the values drastically, but the effect is only to make up for the smaller increases on the short term by larger increases later on. Predictions for things like $300 monthly heating bills in 1985 are not realistic only in that it is clear that homeowners will take strong measures for heat conservation to offset the increased rates.

The cost of collectors is also very approximate, as commercial collectors in our locality vary from $7 per square foot[7] to $25 per square foot. We chose $10 per square foot because that is what we build standard collectors for. (We have developed our own kind of collector, however, at less that half of that.) Clearly the cost per unit area of collectors is one of the factors the student should adjust in making a more detailed study, for even assuming a certain cost per unit area of the collector itself, there must be included the cost of any roof remodeling or construction.

The cost of the plumbing is completely negligible (at least assuming something like a 1500-square-foot ranch style house) in comparison to the cost of the storage tank (installed). Averaging over fiberglass, steel, and concrete storage tanks, we have come up with a figure of $1 per gallon. Clearly this varies enormously with such things as ease of installation. Rock storage for air collectors is roughly the same volume and thus the same cost per Btu of stored heat, so we make no distinction in the kind of storage.

To compare monthly heating costs of the solar versus present heating systems we assume the solar heating system is paid for by a loan. The loan is for ten years with equal payments of the principal plus ten percent per annum interest on the unpaid balance. In addition, we assume five percent per year maintenance costs, and, of course, we add on the fuel cost for the original heating system, now being used for "backup". For example, in a 75 percent solar system there is still 25 percent of the heat being provided by the conventional heating system. Also, it should be kept in mind that the 25 percent heat provided by the backup system is an average over the entire year. During the warm months the house is essentially 100 percent solar heated and during the cold months perhaps only 20 percent to 30 percent solar heated.

After ten years our program does not eliminate the monthly payments on the loan because the fuel costs of the backup overwhelm the effect of such payments.

The Calculations

We now summarize the previous discussion by giving the actual equations we use in our program. To begin with, we calculate the average number of Btu's used per day from the current average monthly heating bill Bo by

$$\begin{aligned} H &= \frac{1000}{30}\left[4000 - \frac{Bo - 5.38}{0.00109}\right] && \text{(natural gas)} \\ &= \frac{3412}{30}\left[271 - \frac{Bo - 7.55}{0.022}\right] && \text{(electricity)} \\ &= \frac{90{,}000}{30}\left[\frac{Bo}{0.49}\right] && \text{(LPG)} \end{aligned}$$

The quantities inside the brackets are the amounts of fuel used per month in cu. ft., KWH, or gal., respectively.

The collector area, in square feet, is now determined by

$$A = \frac{H}{1200},$$

and the number of gallons of storage by

$$\begin{aligned} V &= 0.00188\ H \quad \text{(75\% system)} \\ &= 0.00469\ H \quad \text{(85\% system)} \\ &= 0.00938\ H \quad \text{(95\% system)} \end{aligned}$$

The cost of the solar heating system is then roughly estimated by

$$C = 10\ A + V.$$

The monthly heating bills with no solar heating have, after n years, increased to

$$Bn = (1+r)^n Bo$$

where r is the estimated rate of increase in the fuel costs,

r = 35% for natural gas
= 20% for electricity
= 25% for LPG

After n years, the monthly cost of paying off the loan for the solar heating system, as well as the maintenance and fuel costs for the backup heating system, is given by

$$Pn = \underbrace{\frac{C}{120}}_{\text{equity}} + \underbrace{\frac{i(10-n)C}{120}}_{\text{interest}} + \underbrace{\frac{0.05C}{12}}_{\text{maintenance}} + \underbrace{(1-s)Bn}_{\text{fuel cost}}$$

where i = interest rate on loan (10%)
s = fraction of heat provided by solar heating (e.g., 0.75 for a 75% system)

For the illustration given in Figure 1, where the student had an electrically heated house requiring $60 average monthly fuel cost, the collector area for the solar heating system is 236 sq. ft, the cost of a 75 percent system is $2927, an 85 percent system $3775, and a 95 percent system $5192. His average monthly cost for the all-electric heating will have increased to $200 per month by 1984, while the 75 percent solar heating system will have just increased to his current monthly bill of $60.

Conclusions

Not only several dozen of our students, but several hundred of the general public have been exposed to the computer program described in this paper. There has been great enthusiasm, and, of course, great concern over future heating bills, but two other results have been particularly apparent. First, people have been impressed and made aware of the fact that the heating bill is the best indicator of solar heating requirements. Second, people have been made aware of the fact that even if they want a solar heating system, the first step is to take whatever measures are necessary to reduce their heating bills.

Footnotes

1. Recent prices quoted by the Colorado Springs Department of Public Utilities and local suppliers of LPG.
2. Colorado Springs Public Utilities claims its natural gas varies from 900 to 1200 Btu per cu. ft.
3. This conversion can be found in any physics book.
4. Heat content of LPG quoted by Empire Gas Co. in Colorado Springs.
5. Collector area requirements are calculated according to Kreider and Kreith, Solar Heating and Cooling, McGraw Hill, 1975.
6. One gallon of water storage can hold 8 Btu's for each degree F above the inside house temperature.
7. The least expensive commercially available collector of which we are aware is made by Northrup Corp., Dallas, Texas, for $7 per square foot.

THE DESIGN AND IMPLEMENTATION OF COMPUTER ASSISTED INSTRUCTIONAL MODULES IN ENGINEERING PHYSICS

by

Grayson H. Walker

Department of Physics

and

Homer B. Utley

Cooperative General Science Project

Clark College

Atlanta, GA 30314

INTRODUCTION

Service courses in subjects such as mathematics, chemistry, and physics form the foundation for the engineering student's future training. However, these vitally important courses are usually the ones that suffer from the efforts of economy minded administrators. While the economic situation existent in many colleges indicates the necessity of increasing the student/teacher ratio, the pedagogical effects of such measures are well known. In response to this situation, many instructors have turned to computer based instructional tools in an effort to improve instruction in classes with large enrollments (1). In this article we discuss a sequence of computerized tutorial-drill modules that have shown promise in providing a degree of individualized instruction and remedial work (if needed) under circumstances that, in the absence of computerization, would almost certainly obviate these teaching methods.

Computer assisted instruction (CAI) has been widely used in a number of guises in a variety of physics courses ranging from elementary, introductory type courses to advanced upper level, or even graduate level courses. The amount of CAI actually incorporated into the course and the overall impact of CAI on the course also varies widely: some instructors use CAI as a supplement or possibly in a remedial context while others (see Reference 2 for a typical example) use

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

it to affect a drastic alteration of the course in a way not possible without extensive computational facilities. Based upon the resources available and upon the needs of the particular courses involved, we chose an intermediate approach: CAI counted as an important portion of the course yet it was to be fitted into the traditional flow of the instructional materials. The programs developed in this work were written in BASIC and were designed to incorporate a large degree of transportability with a minimal amount of modification (3).

THE ACADEMIC ENVIRONMENT

Clark College is a four year, liberal arts school having an enrollment of about 1800 students. It offers degrees in a wide variety of areas and is a member of the Atlanta University Center, a consortium of six geographically adajacent predominately Black colleges whose total enrollment is nearly 6000 students. While none of the colleges in the Center offer engineering degrees, there is a very active pre-engineering program that functions in cooperation with a number of nationally prominent engineering schools. Entrance requirements and the degrees of student preparedness vary drastically throughout the Center and, as a consequence, programs that draw students from all the colleges must deal with an extremely heterogeneous population.

All pre-engineering students are required to take a three semester sequence of physics courses (Physics I, II, and III). There are three lecture and three laboratory hours per week in each course. A beginning course in calculus is a corequisite and calculus is used freely throughout the three courses. In many respects these courses are similiar to courses found in most universities, the major difference being that almost all the students in them will eventually go into some phase of engineering. For this reason the subject material covered is given a decidedly engineering slant, particularly in the written assignments and on tests; however, the physicist's emphasis upon the understanding of natural laws is maintained.

Physics III, the last course in the sequence dealing with vibrations, wave motion, optics, and modern physics, was chosen as the one for testing computer assisted instruction. The students in this class have survived two previous semesters of physics with the same instructor and represent a mature and fairly homogeneous population. The typical class at this level is small - roughly thirty students. Hence we elected to use this course rather than the large, academically heterogeneous classes normally found in Physics I and II since this would give us a target population of known characteristics. Ultimately, we intend to use some form of CAI in all three courses.

A set of six modules were developed, each module containing related materials. Topics covered include (a) free, damped and forced vibrations, (b) wave motion (both sound and electromagnetic waves), (c) reflection and refraction, (d) geometric optics, (e) interference and diffraction, and (f) elementary quantum physics. These modules closely parallel lecture materials and laboratory and written assignments.

EDUCATIONAL STRATEGY

Before choosing the form in which CAI would be introduced into Physics III, one of us (GHW) discussed the proposed project extensively with students during the pre-registeration period. We sought to ascertain exactly what students expected in an ideal learning situation and how courses they had taken deviated from this. Their responses fell into four broad categories which we may summarize as:

1. Immediate responses to the correctness of the attempted solutions to homework problems.
2. Occasional hints on how to attack specific problems;
3. A "pressure free" learning environment with a large measure of self pacing; and

4. Effective utilization of instructional time and an avoidance of repetition.

In planning the educational strategy used in our CAI programs we attempted to utilize these points in so far as possible.

The standard tutorial method was chosen: the student is presented a question or problem that either parallels the examples in the textbook or is closely related to examples from the lecture. The parameters in the problem are produced using subroutines based upon a pseudorandom number generator. The student then responds with an answer and is notified whether this answer is correct (within some margin of accuracy, usually 0.1%). If the answer is incorrect, the student attempts the problem again with no grade penalty. A second wrong response produces a mild hint from the computer. The student tries again. A third wrong response results in a much stronger hint to the student while a fourth incorrect answer results in a "strike out" after which the student must proceed to the next question. Students reaching this stage on any problem were given special attention by the instructor.

During the four possible attempts on a given problem, the problem parameters are unchanged. A correct response on the first or second trial gives the student full credit. A correct response on the third trial gives 75% credit and a correct response on the fourth trial gives 50% credit. As a bonus, the student who answers the question correctly on the third or fourth trial receives a "redemption question." The redemption question or problem is the same problem with changed parameters. A correct response at this point results in full credit for the exercise. The student is then expected to go on to the next problem.

The rationale upon which this somewhat complicated proceedure is based lies in the four points listed above. By giving two penalty free trials and the redemption problem, performance anxiety was reduced, a point verified by responses on a student evaluation. By allowing a number of trials, a student's performance on a given module could be accepted without the need for a separate test on the module. Alternate schemes, where the students work on the tutorial as much as they like and then take a computerized examination one or several times seemed to us to be a less efficient use of a student's already limited time than the scheme utilized in our programs.

Each module contained a set of forty problems divided into ten sections of four, closely related, problems. The problems ranged from easy, review-type questions to some very difficult ones located near the end of the module. The student was expected to proceed sequentially through the module although the module itself could be completed over a number of sessions. There was, however, some element of freedom in choosing a starting section and question at each session.

THE CAI PACKAGE

The computerized tutorial described in the previous section was programmed in BASIC PLUS, the Digital Equipment Corporation's version of the widely used BASIC language. Provisions can be made to adapt the package to any reasonably complete subset of standard BASIC (3). The entire package described in this section has been used extensively over a six month period on a DEC [illegible]/40 system operating under the RSTS/E operating package supplied by DEC. and [illegible] proved to be reliable and reasonably efficient in terms of processor time and overhead.

The package consists of two programs, ROLL and PHYS3, which, in conjunction with the system's text editor, suffice to handle all bookkeeping chores, create question text files, and handle the flow of the tutorial sessions. We now describe briefly their function; more extensive detail may be obtained from Reference 3.

A. ROLL

Since the entire package will ultimately be used in a number of large classes and since each student will have his own file which will record his progress, his responses, the date and times of CAI sessions, and ulitmately his grade for the CAI segment of the course, a program was needed that would enable the user to key in the name of his class, enter each student's name, and then assign a unique file name to each student. ROLL performs these tasks as well as preliminary initialization of each student's file. ROLL can also compute each student's final average and give a summary of the frequency of CAI usage and time spent on each module.

B. PHYS3.

PHYS3 handles the tasks of directing the tutorial session, reading question files, generating problem parameters, checking student answers and updating the student's file. It is a lengthy program requiring about 16K words of memory; however, it is easily chained into two shorter programs.

Ignoring the bookkeeping tasks, the main function of PHYS3 is to read a problem file corresponding to a pre-selected module, generate the problem parameters, calculate the correct solutions, and to compare these solutions with those presented by the student. This is accomplished as follows: First a set of files, one for each module, containing the problems was generated using the system's text editor. The use of an editor is not mandatory - systems not having one always possess alternate methods for generating such files. In the problems instead of inserting numerical values for the parameters, the # sign was used as a place marker, one # sign for each position needed to display the parameter. Provisions were made in the program for the insertion of up to ten parameters, more than adequate for all the problems in this level of course. The parameters were generated by a user-defined function whose arguments included upper and lower limits and the number of digits to the right of the decimal place for a particular parameter. When the problem was displayed, these values were inserted appropriately into the problem text.

Two additional text files were also created. These contain two levels of hints or suggestions for each problem. PHYS3 refers to these files upon request from the user or automatically on the third and fourth trials for a given problem.

It was found convenient to write a separate subroutine for each question. These subroutines initialize the problem parameters, cause the question to be displayed, and calculate the correct answer. Usually these subroutines are short, typically five to ten lines in length, and are easy to write. They are also easy to change, an important point, since each module requires its own unique subroutine package. At present, the programming of these subroutines is the major task facing the would-be user of PHYS3. Alternate and possibly less time consuming schemes are being investigated.

EVALUATION OF THE SYSTEM AND ITS IMPACT

Student response to and the use of the system was generally excellent. A course evaluation questionnaire revealed that all the students had a very positive view of CAI; however, none of the students had had prior experience with CAI and thus some of the favorable response is no doubt due to the novelty of the idea. The major negative criticism was directed to the wording of the questions, hints, and suggestions, and to certain inconveniences related to terminal devices and saturation of the time sharing system around the end of the semester.

The use of the modules had two very noticable effects: first, two weeks of additional materials were covered in the CAI class as compared to an equivalent non-CAI class and, second, the average grade on the materials covered in the set of CAI modules was almost one letter grade higher than in the non-CAI class. Ultimately we plan to test the intermediate range effect of CAI on retentiveness by introducing CAI in the first semester (Physics I) and then testing the concepts learned in the CAI modules after an elapsed time of 12 to 15 months. The results of these tests will be compared with the performances of an equivalent but non-CAI class on identical materials.

Students used the terminals at an average rate of about four hours per module and averaged slightly more than one minute of CPU time (on a PDP 11/40 with RSTS/E operating system) per hour of connect time. Since there were a total of six modules, the total connect time per student was around 24 hours with a total CPU time of about 25 minutes. As is to be expected, most of the time was logged by weaker students who, it seemed, benefited the most from the CAI.

CONCLUSIONS

In the preceding sections we have described a computer assisted teaching procedure and its accompanying software. The hardware requirements are modest - most minicomputer systems can support it. It is easily transportable in that only minor and easily located modifications are needed to move it to a new environment. And, while it was written with the needs of engineering students in mind, it is widely applicable to other disciplines with a problem solving orientation.

Based upon two semesters of use, the approach and materials used have been successful. Current work is centered upon improving the problems and related materials and upon adding a method of detecting consistently made errors.

ACKNOWLEDGMENTS

We wish to express gratitude to Mr. Grover Simmons, Director of the AUC Computation Center, to Dr. John Hall of Morehouse College, and to Mr. Beauregard Hardemann for their help during the 1977 NSF Summer Workshop where this work was begun.

REFERENCES

1. Hammer, Marck and C. Obert Henderson, "Improving Large Enrollment Undergraduate Instruction with Computer Generated, Repeatable Tests," in Proceedings of the 1972 Conference on Computers in Undergraduate Curricula, distributed by the Southern Regional Education Board, Atlanta, GA 30313, p. 209.
2. Bork, Alfred and Joseph Marasco, "Modes of Computer Usuage in Science," T. H. E. Journal, Vol. 4, No. 2, Feb. 1977.
3. Further documentation may be obtained from G. H. Walker, Department of Physics Clark College, Atlanta, GA 30314.

POPULATION DYNAMICS WITH A COMPUTER

Robert P. Banaugh

University of Montana

Abstract

By using examples from population dynamics, it is demonstrated that, by bypassing the language of mathematics and proceeding directly in a programming language, it is possible to obtain a ready analysis of quantitative biological phenomena. Experience in using this direct computer based approach with bioscience students for over five years confirms the ease with which the approach is understood by the students. In addition, in contrast to a traditional mathematically based approach, far more class time may be spent on developing and discussing the relevant biological material.

Introduction

It is usual practice to formulate and analyze problems in population dynamics with the aid of the continuous calculus. This results in a set of differential and/or integral equations which must be solved to permit the analysis. Frequently, it is then necessary to devote a large amount of class time (which amount of time usually far exceeds that given to discussing the formulation of the problem or to discussing the results of the analysis) to developing a method of solution of the equations. Sometimes the equations are intractable and then class time is required for the development of mathematical techniques for determining properties of the solution without actually being able to find the solution. In either event it is usually the case that the bioscience student loses sight of the 'biological' forest by becoming lost in the 'mathematical' trees. Using examples, this paper will show that, by expressing directly in a programming language quantitative hypotheses which the investigator assumes govern the behavior of the phenomenon under consideration, very straightforward and readily understood analysis can often be obtained. In such a formulation of the problem, the language of mathematics is effectively bypassed. The programming language chosen was BASIC because it is interactive, understood by many students, very easy to learn and readily available on all sizes of computers.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

Examples and Discussion

As a first example, consider the simple problem of the growth of a population in a constant environment. This is usually called the Malthus problem. The analysis of this problem, as well as subsequent problems, will be based upon the fundamental law of change which is expressed in BASIC as

$$V(I+1) = V(I) + C(I) \qquad (1)$$

where $V(I+1)$ and $V(I)$ are the new and old states of the variable respectively, $C(I)$ represents the change in the variable during a time period and the index I counts the time periods which are assumed to be constant. The relation of this equation to the differential equation formulation,

$$\frac{dv}{dt} = c(v,t), \qquad (2)$$

is evident if a first order difference approximation to the derivative is made and the time increment in the difference approximation is chosen to be the time period.

For the Malthus problem it is usual to consider a population growing in an environment in which there is no immigration nor emigration and thus the change in the population from one time period to the next is the difference between the number of births and the number of deaths occurring during the time period. It is also assumed that, for each member of the population, the environment remains constant. Thus, such population parameters as birth and mortality rate remain constant during the evolution of the population. In light of these assumptions it is reasonable to state that the number of births and the number of deaths during a time period should each be proportional to the population existing at the beginning of the time period. With B and M denoting the respective constants of proportionality, the fundamental equation is then

$$P(I+1) = P(I) + B*P(I) - M*P(I) \quad . \qquad (3)$$

The magnitudes of B and M depend upon the length of the time period since they relate the population to the respective number of births and deaths during the time period.

By inserting equation (3) into a loop over the index I, making provision for entering the initial population P(0) together with the growth coefficients, B and M, and finally adding the appropriate output statements to print out the calculated results, a simple 5 or 6 line program is obtained. This program runs in a very short time and thus a few quick runs using different values of B, M, and P(0) permit ready insights concerning the variation in the population as the three parameters are varied. Equation (3) is not put into a more simplified form since, for such a simple problem, the reduction in calculation is not significant and it is much easier to modify the equation when it remains in expanded form. If the birth rate exceeds the mortality rate the numerical results will indicate exponential growth and if the mortality rate exceeds the birth rate an exponential decrease in the population will be indicated.

The program is easily modified to account for the effect of a finite environment. It is assumed that, for a population existing in a finite environment, as the population increases the amount of resources available to each individual decreases. Hence an increasing population results in a decreasing birth rate and an increasing mortality rate. The simplest alteration of the Malthus model to accommodate this hypothesis is to decrease B by an amount proportional to the population and to increase M by an amount proportional to the population. Thus B is replaced by the expression (B-B1*P(I)) and M is replaced by the expression (M+M1*P(I)) where B1 and M1 are constants of proportionality and are commonly called the auxiliary birth and mortality coefficients respectively. Both are positive and very much less than B and M respectively.

Replacing B and M in equation (3) by the respective terms B-B1*P(I) and M+M1*P(I), and altering the input statement to provide for the entry of the auxiliary birth and mortality coefficients, gives a program which describes a population growing in an environment whose resources are limited. Equation (3) becomes

$$P(I+1) = P(I) + (B-B1*P(I))*P(I) - (M+M1*P(I))*P(I) \; . \qquad (4)$$

The condition that there be no change in the population during a time period is equivalent to setting

$$P(I+1) = P(I) \qquad (5)$$

This condition, coupled with equation (4), states that the value of the population for which there is no change in the population from one period to the next is

$$\overline{P} = \frac{B - M}{B1+M1} \qquad (6)$$

$\overline{P}$ is called the carrying capacity and this expression is usually obtained by a similar analysis of the problem as it is formulated in the language of differential equations. Since it is common to denote B-M by G, and B1 + M1 by G1, where G and G1 are called the growth coefficient and the auxiliary growth coefficient respectively, it follows that the carrying capacity is

$$\overline{P} = G/G1 \qquad (7)$$

Equation (4) can be easily further modified to describe a population growing in an environment in which contamination created by the growth of the population affects the growth of the population. This is accomplished by assuming that the effect of the intoxicants created by the growing population will be such as to further decrease the birth rate and increase the mortality rate. In order to impose this assumption on equation (4), let T(I) denote the amount of intoxicant created by the population up to the beginning of the I^{th} time period. To estimate T(I) it is assumed that the amount of intoxication created in a time period is proportional to the population during the period. If C denotes the contamination constant of proportionality, then the amount of contamination created in the I^{th} time period is C*P(I-1). Since the amount of intoxicant present in the I^{th} time period is the sum of the amounts of intoxicant produced in the previous time periods, it follows that

$$T(I) = C*P(0)+C*P(1)+C*P(2)+ \ldots\ldots +C*P(I-1) \qquad (8)$$

This is the desired expression for $T(I)$.

The expressions for the modified birth and mortality rates are obtained by assuming that the birth rate during a time period will be decreased in proportion to the amount of contamination in existence at the beginning of the time period and the mortality rate will be increased in an analogous manner. The effect of contamination is then accounted for in the model by altering the proportions of births and deaths during a time period to read

$$B-B1*P(I)-B2*T(I) \quad \text{and} \quad M+M1*P(I)+M2*T(I),$$

where $B2$ and $M2$ are constants of proportionality, positive and very much less than B and M respectively. If the notation $G2=B2+M2$ is introduced, the equation describing the growth of a population in a limited environment and in the presence of contamination is

$$P(I+1) = P(I) + G*P(I) - G1*P(I)\uparrow 2 - G2*T(I)*P(I) \ . \qquad (9)$$

In this form, an examination of the equation reveals that the term $G*P(I)$ accounts for the growth in a constant environment, the term $-G1*P(I)\uparrow 2$ accounts for the effect of limited resources and the last term, $-G2*T(I)*P(I)$, represents the effect of contamination on the population growth.

Figure (1) illustrates results of a program describing the growth of a population in which the birth and mortality rates remain constant. Two cases of interest are portrayed. The solid line indicates a run for which the birth rate exceeds the mortality rate and the dashed line indicates a run for which the reverse is true. As expected, the population exhibits an indefinite exponential increase when a constant birth rate exceeds a constant mortality rate. When the reverse is true, the population dies out as the dashed line shows. Figure (2) illustrates a run obtained using equation (4). The behavior of the population is characterized by the familiar 'S' curve and the limiting value attained by the population is indeed the quotient of the two numerical values used for G and G1 respectively in the computer run. Finally, figure (3) portrays a result obtained by imposing the effect of contamination on the previous model. Both equations (8) and (9) were used to obtain this result. It is seen that the population initially rises exponentially and then as the growth restraint of a limited environment is approached, the population reaches a maximum. At this point the toxicity causes the population to start to decrease and the continual increase of the toxicity eventually causes the population to die out. Because the toxicity is cumulative, the toxicity curve continues to rise as long as the population exists. However, as the population diminishes, the growth rate of the toxicity curve decreases, and hence the curve also resembles an 'S' curve. It is possible to modify the contamination model still further by including retarded time effects. Such a modification permits a computer based study of biodegradable contaminants. It also enables a study of the known increasing ability of successive generations of a population to neutralize an intoxicant. This modification has been made by the author and the results are in accord with observation.

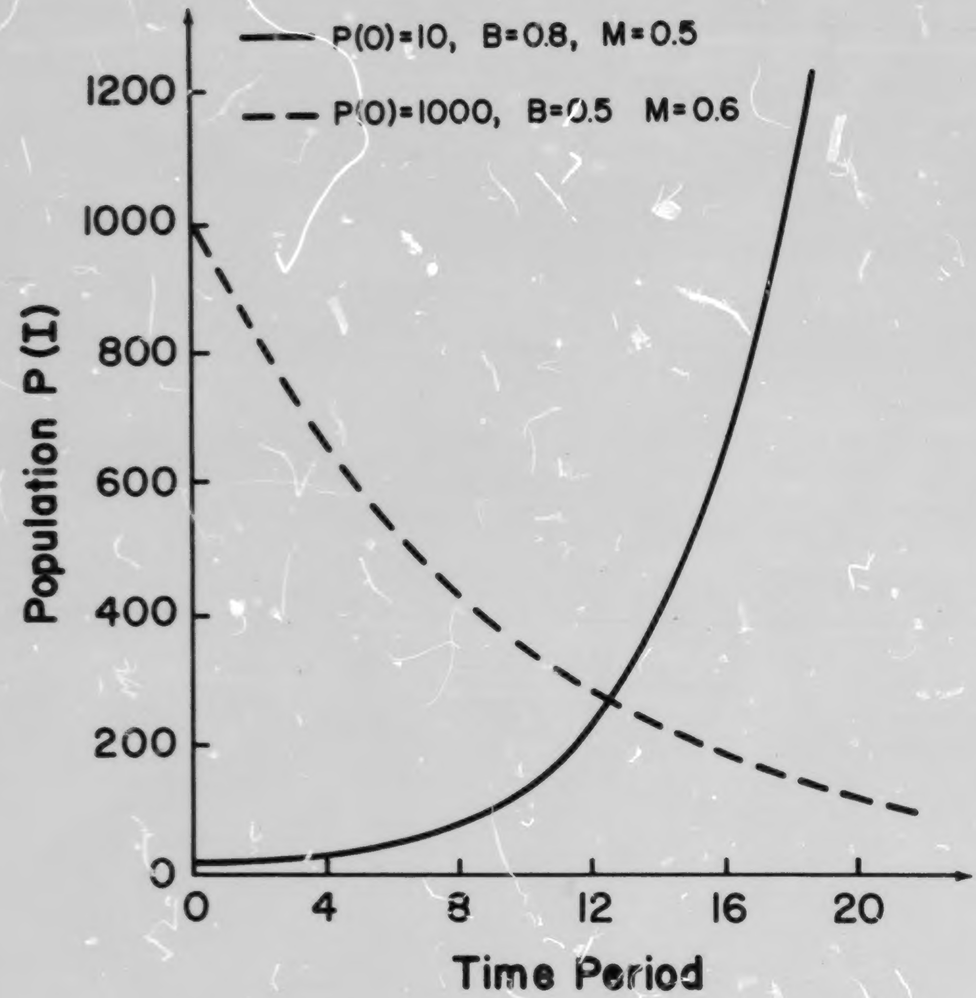


Fig. 1 Constant Environment Model

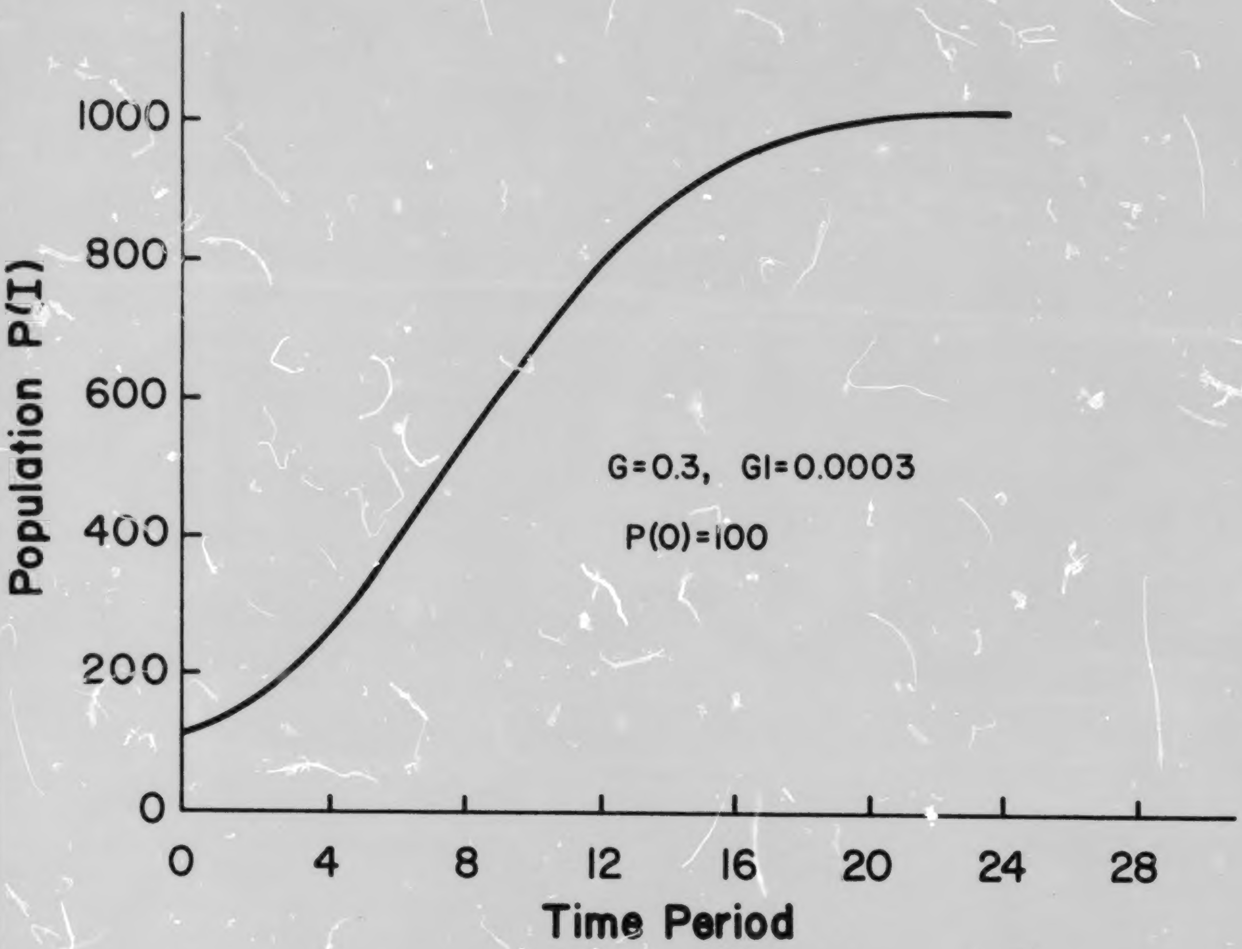


Fig. 2 Limited Environment Model

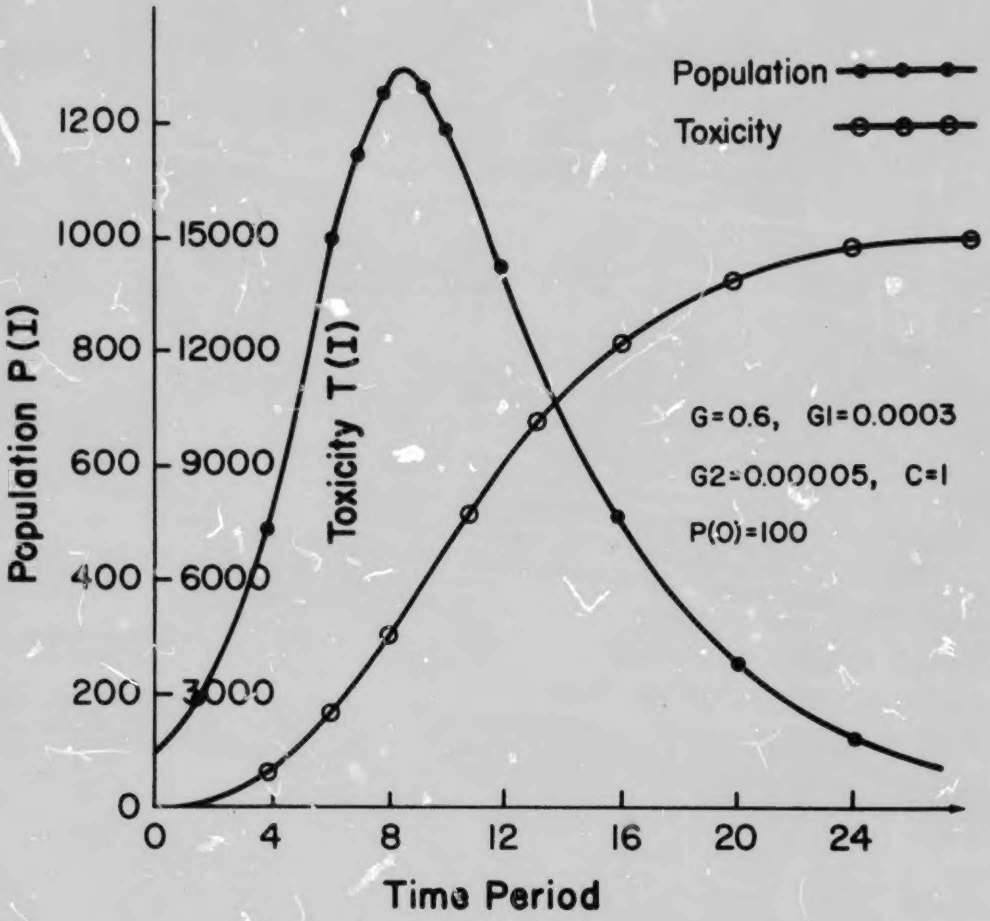


Fig. 3 Contaminated Environment Model

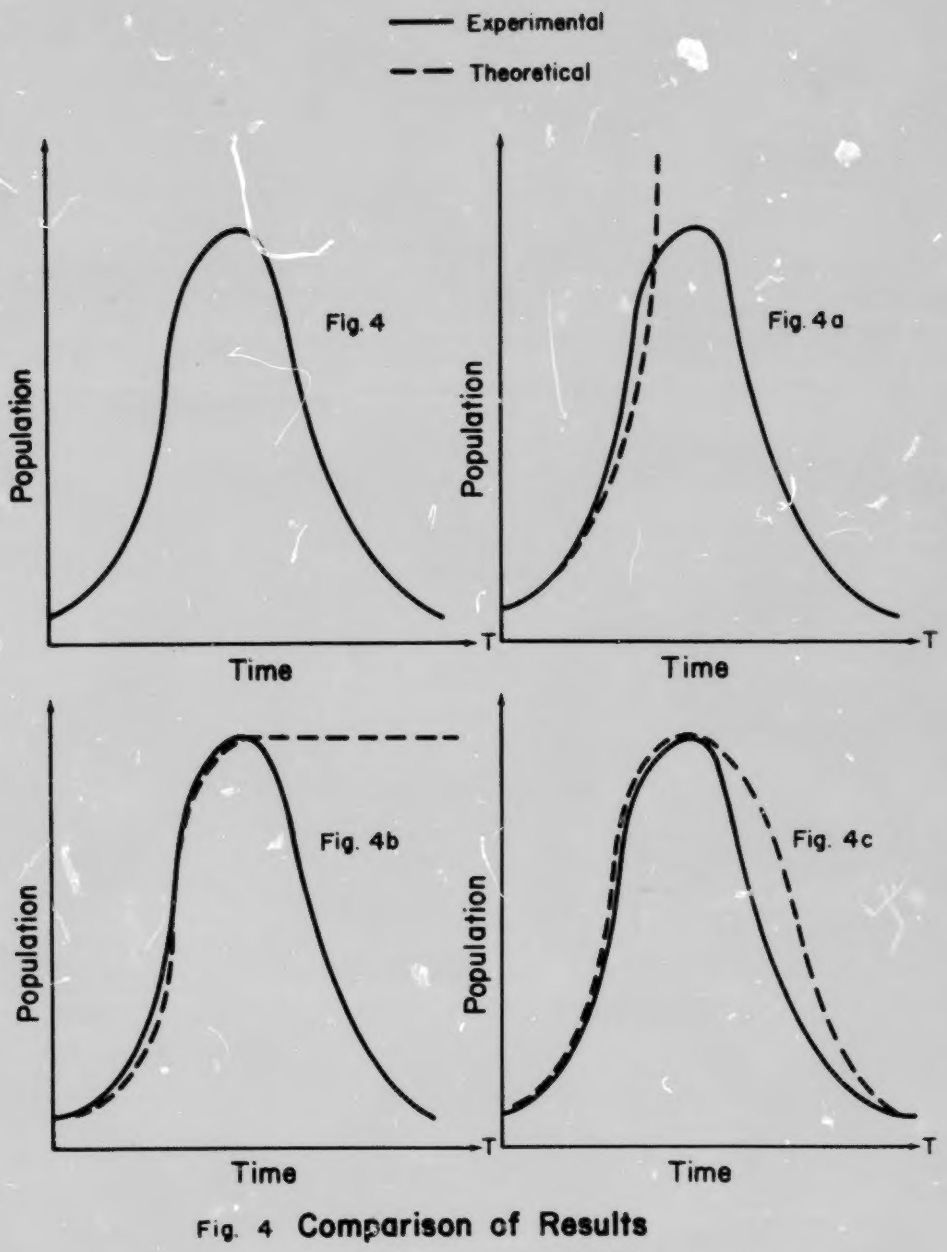


Fig. 4 Comparison of Results

It is interesting to compare the results obtained from each of the three models. Figure (4) shows a population growth curve of a bacterial culture and is taken from the work of Buchanan and Fuller in 1928 as reported in D'Ancona (1954). Figure (4a) is a comparison of the results obtained from the Malthus model using a positive growth coefficient. It is seen that the early time behavior of the curves are quite similar. Consequently, it may be inferred that the initial growth of a bacterial culture is exponential in character. Figure (4b) is a comparison of the same experimental data with numerical results obtained from the finite resource model. The two curves compare quite well in shape and form up to maximum growth. For early time periods, the finite resource model agrees just as closely as does the constant environment model results. However, since the finite resource model also agrees for a longer period of time, we conclude that the latter model provides a better description of bacterial culture growth than does the former model. Finally, figure (4b) presents a comparison with the results obtained from the contamination model. It is seen that, compared to the results of the other two models, the contamination model yielded results which were in longer time agreement. Hence, one may conclude that this model is an even better description of the bacterial growth.

An Extension

It is possible to alter the expression for T(I) as described by equation (8) to allow for the fact that the effect of the contamination varies according to the time of existence of the contaminant. Thus, intoxicants created early in the history of the culture can have a different effect on the population growth than intoxicants recently created. One technique for accomplishing this is to write

```
200 LET T(I)=0

210 FOR J=0 TO I

220 LET T(I)=T(I) + C*D(I-J)*P(J)

230 NEXT J
```

The statement numbers serve only to indicate the order of the operations and bear no relation to the place in the program of the above set of operations. The calculation specified by these operations is to be carried out each time period while advancing the population from time period I to time period (I+1). The term D(I-J) is a multiplier accounting for the potency of the contaminant relative to the potency of an equal amount of contaminant which has just been created. The obtaining of such retarded time coefficients is usually a difficult task.

The above calculation for T(I) is a first order discrete equivalent for the numerical evaluation of the retarded time integral

$$C \int_{\tau = 0}^{\tau = t} D(t-\tau)P(\tau)d\tau$$

Hence, the use of this calculation for T(I) in the population growth model would be, in effect, numerically solving the integro-differential equation

$$\frac{dP}{dt} = \left[G - G1\ P(t) - C \int_{\tau = 0}^{\tau = t} D(t-\tau) P(\tau) d\tau \right] P(t)$$

The mathematical study of such equations is usually only attempted in graduate level mathematics courses and if the term $D(t-\tau)$ is not exceedingly simple in form, the solutions of such equations must be effected by numerical methods. Thus, even a mathematically based discussion of the growth of a single population growing in a contaminated environment would necessitate the writing of a program very similar to that which has been described above. This then, is an example of the power of directly formulating and thinking about quantitative phenomena in terms of a higher level programming language such as BASIC.

Pedagogical Implications

The ease with which this sequence of models was constructed, and the consequent numerical results obtained and compared with empirical data, is evident. Students with no knowledge of the calculus and/or differential equations, can very readily understand and construct such computer based models. In addition, during the development of the programs, the attention of the student is properly focused on the determination of biological hypotheses concerning the phenomenon under consideration and not on attaining techniques for the solution of specific equations. In contrast, the traditional approach based upon the calculus, etc. requires that the student concentrate not on the underlying biological ideas, but rather on acquiring mathematical techniques for the solution of a particular problem. The direct computer based approach means that more of the learning effort of the student can be devoted to discussing quantitative hypotheses and exploring the consequences of these hypotheses. The author and his colleague, W. R. Pierce of the University of Montana, School of Forestry, have jointly taught a course based on the technique of formulating directly in the BASIC programming language quantitative hypotheses describing biological phenomenon. The course is designed for sophomores in the biosciences and in the School of Forestry and has proved very popular. The course covers such topics as: the interaction of several populations, life table analyses, population genetics, compartmental analyses, etc. In none of these topics is a knowledge of the calculus, differential equations, matrix algebra, mathematical probability, etc. used or assumed. The authors would be glad to share the work with faculty members interested in the pedagogical problem of enabling bioscience students to become aware of, and familiar with, quantitative methods in the biosciences.

It is not evident that the use of a computer as a tool to carry out computations originating from a mathematically posed problem is the only way to teach students quantitative methods. It may not even be the best way. The intent of the authors' work is to explore the bypassing of the language of mathematics by proceeding directly in an easily understood language. Students are "turned on" by computers and this desire can be used to assist the student's learning process. Sometimes sophisticated mathematical analysis may be required to attain an understanding of a problem. However, it is more frequently the case that the use of a digital computer is the only way an analysis can ever be obtained. In conclusion, the fact that the digital computer provides all students with an increase in arithmetic capability of more than ten orders of magnitude has yet to begin to impact pedagogical techniques in the sciences. The authors' work is an attempt to begin to utilize this tremendously increased arithmetic capability.

This work was sponsored in part by National Science Foundation Grant Number GZ 3469.

Reference

1. Banaugh, R. P. Computer Based Methods in the Biological Sciences, An Introduction. 680pp. 1976. Class Notes.

MINICOMPUTER SYSTEMS AS A VEHICLE FOR TEACHING COMPUTER AND OPERATING SYSTEMS

by Dr. Emil C. Neu
Department of Electrical Engineering
Stevens Institute of Technology
Castle Point Station
Hoboken, New Jersey 07030
(201) 792-2700, Ext. 263

INTRODUCTION

An important component of any undergraduate curriculum in computer science is a course or a sequence of courses dealing with computer and operating systems.[1] This paper describes the implementation of such a course with a DEC PDP11/40 minicomputer system.

The course assumes a background in computer organization, programming and data structures but no experience in the actual operation of a machine. The lectures in this course deal with operating systems in general, and whenever possible, the PDP11/40 system in particular. These lectures are complemented by laboratory projects in which the students are expected to be involved in all aspects of programming and machine operation.

SYSTEM CONFIGURATION

The original PDP11/40 minicomputer system in the Department of Electrical Engineering was obtained in July of 1973. This system was based upon paper tape as the bulk storage medium. Major hardware, besides the processor itself, included 8K of 16-bit core memory; a PC11 high-speed paper tape reader/punch; and an LA30-P DEC writer terminal. Paper tape software included the standard utility programs such as an absolute loader, editor, assembler, on-line debugger and a single user BASIC interpreter. In addition, applications programs were obtained from DECUS* (a clearing house for user-developed programs) and subsequently some programs were developed at Stevens.

---

*Digital Equipment Users Society, Maynard, Massachusetts, 01754

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

This system had limited capability, principally because of the small amount of core and slow speed of the paper tape as compared to other bulk storage media. The core size meant that only one operating program could reside in memory at one time and thus programs had to be loaded frequently via paper tape.

In September of 1977, with the aid of a grant from the National Science Foundation, the system underwent a major expansion. This expansion included the purchase of 48K of core (for a total of 56K of core), two disk drives, a magnetic tape drive, a dual floppy disk drive and two I/O ports (for a total of three I/O ports). In addition, the UNIX[12] operating system was obtained from Bell Telephone Laboratories.

The disk system provides expanded storage facilities which permit quick access to programs and data. This eliminates the time-consuming task of using a number of paper tapes to load and execute even a small program. The magnetic tape system provides for backup copies of the disks, as well as copies of special purpose utility programs. In addition, the magnetic tapes are compatible with PDP10 system on campus. This facilitates the transport of programs such as assemblers and the use of the PAL11 simulator on the PDP10. (PAL11 is the assembly language for the PDP11).

The floppy disk system provides an inexpensive random access medium for the storage of user files. This is particularly advantageous in the case of an undergraduate course since each student can be assigned his or her own floppy disk.

The UNIX operating system controls the disk, magnetic tape and floppy disk systems. UNIX is a time-sharing system and presently handles a DECwriter and two teletypes (the acquisition of two video displays and a line-printer is under consideration). In addition to providing quick and easy preparation, translation, linking, debugging and execution of user programs, UNIX also provides for file and library maintenance. A variety of languages such as PAL11, BASIC and Fortran are available on the system, as well as special functions such as type setting and a desk calculator.

The selection of the PDP11 system, as opposed to other minicomputer systems, has proven to be a wise choice. Its wide use has resulted in the publication of numerous textbooks and manuals[3, 6, 13, 14] based on the system, in addition to the manufacturer's literature. Furthermore, components and peripherals are available from a variety of sources, as well as the manufacturer.

The UNIX operating system, which was developed specifically for the PDP11, was selected in preference to the manufacturer's system. UNIX is well-documented by information supplied with the system[2, 15] and by materials published by users.[7,8] In addition, there is a UNIX user's group*, which facilitates the exchange of information between users.

As an operating system UNIX is well-suited for use in an undergraduate course. While it is a sophisticated operating system, it is compact in terms of the amount of code involved. Furthermore, it is accessible which makes its study and even its modification feasible.

---

*UNIX User's Group, Professor Melvin Ferentz, Physics Department Brooklyn College of CUNY, Brooklyn, N.Y. 11210

PRIMITIVE OPERATING SYSTEMS

As an introduction to operating systems and to establish the need for an operating system, primitive operating systems are discussed. In these systems the user is the programmer, operator and operating system.

The students are given a hands-on experience on such a system by requiring them to create and run a PAL11 program using only the CPU, 8K of core, the paper tape reader/punch and the DECwriter. Furthermore, core is assumed to be initially clear of any programs. The procedure is as follows:

1. Toggle in the bootstrap loader.
2. Use the bootstrap loader to load the absolute loader, which is on paper tape.
3. Use the absolute loader to load the editor, which is on paper tape.
4. Use the editor to create a file containing the source program.
5. Output the source program on paper tape.
6. Use the absolute loader to load the PAL11 assembler, which is on paper tape.
7. Read the source program paper tape twice, corresponding to the two passes of the assembler.
8. Output the object program on paper tape.
9. Read the object program paper tape to execute the program.
10. If any changes are to be made in the program, the above procedure is repeated, starting with step 3.

This time consuming process dramatically impresses upon the students the need for an operating system. The effect is even more impressive if they should accidentally wipe-out the absolute loader and bootstrap, in which case the entire process must be repeated, starting with the toggling of the bootstrap.

ASSEMBLY LANGUAGE PROGRAMMING

Since a knowledge of the assembly language for the PDP11 is a pre-requisite for subsequent topics, an introduction to PAL11 is presented at this point in the course. The presentation is facilitated by the fact that the students already are familiar with an assembly language; namely, Macro-10, the assembly language for the PDP10. Aside from the differences in the instruction set from Macro-10, there are certain aspects of PAL11 which require careful consideration. These include the numerous addressing modes and the fact that in certain instances instructions are stored in multiple words.[5]

The classroom discussion of PAL11 is complemented by a PAL11 programming project. This project not only serves to reinforce the students' understanding of PAL11 but also provides an introduction to the UNIX operating system.

Specifically the following aspects of UNIX are involved in this project:

1. Booting UNIX.
2. Mounting the disks (physically and softwarewise).
3. Setting the date.
4. Inserting the user name and password into the password file.
5. Logging in.
6. Identification of the differences between DEC PAL11 and UNIX PAL11.[11]
7. Use of the UNIX text editor.
8. Formatting a floppy disk.
9. Mounting a floppy disk.
10. Use of a floppy disk.
11. Assembly of a PAL11 program.
12. Execution of a PAL11 program.

It should be noted that this use of UNIX leads to a familiarity with the operating system which is valuable in its subsequent theoretical study.

C-LANGUAGE

UNIX is written in a language called C, developed at Bell Telephone Laboratories.[10] In order to facilitate the study of certain details of UNIX, an introductory treatment of C is presented. The classroom discussion of C is reinforced with a programming project.

It should be noted that C is a modern, well-structured language with capabilities not found in languages such as Fortran. Thus it is worth studying aside from its usefulness in understanding UNIX.

LOADERS, ASSEMBLERS AND EDITORS

Previously, in the course the students have had an opportunity to work with three types of system programs on paper tape; namely, loaders, assemblers and editors. The fact that these programs are on paper tape emphasizes the fact that these are indeed programs and not some mysterious force within the computer.

First the concept of bootstrapping is discussed. This is done with reference to the experience the students already have had in starting with a very simple loader, which can be toggled in, and using it to read a more complex loader. The previously used bootstrap program is available for study in this connection.

This discussion is followed by a study of absolute loaders. The absolute loader used in conjunction with the bootstrapping process is cited as an example of this type of loader.

Linking loaders are considered next. Topics studied include static relocation, the chaining method of linking and the transfer vector method of linking. The linking and loading tasks are performed by a link editor and a loader in UNIX. Of these two programs, the link editor is available to the user. Hence this program is studied as a specific example of a linker. This is followed-up with a programming project involving the linking of a program and a subprogram in separate files.

Assemblers also are studied at this time. The basic techniques[4] involved in various types of assemblers are presented with the emphasis on two pass assemblers. This discussion is complemented with a study of certain aspects of the PAL11 assembler.

Editors are mentioned at this point in the course since the students have utilized two editor programs. Furthermore, editors contain some of the same symbol manipulation techniques utilized in assemblers.

I/O PROGRAMMING

One of the problems encountered in early computer systems stemmed from the disparity between CPU and input/output device speeds. Specifically, early systems wasted CPU time by using it to control the relatively slow I/O devices.

Various solutions to this problem are discussed. These include channels and channel programs, buffering and interrupts.

I/O programming in PAL11 is studied under the assumption of no operating system. A programming project involving device testing and interrupt programming is assigned.

In UNIX I/O is handled by software device drivers in conjunction with buffers and interrupts. The study of these mechanisms is reinforced with a programming project involving a device software driver.

MEMORY MANAGEMENT

Various memory management schemes are studied starting with relatively simple schemes and concluding with schemes in use in large multi-programming systems. Specifically included are single contiguous allocation, partitioned allocation, relocatable partitioned memory management, paged memory management, demand-paged memory management, segmented memory management and segmented and demand-paged memory management.

Since our PDP11/40 contains more core than can be addressed by the 16 bit words of this machine, the memory management option was purchased. This option extends the addressing capability by permitting 18 bit address specification. As a project the students are asked to experiment with storing and running of programs in various areas of core.

PROCESSOR MANAGEMENT

A general discussion of processor management includes such topics as the state model, job scheduling, process scheduling, multiprocessor systems and process synchronization. A description of UNIX process management also is given.

As a project the students are asked to run a foreground program and a background program using UNIX. It might be observed that this project is of particular practical significance since the present machine configuration lacks a floating point unit and thus executes Fortran programs very slowly. Consequently, it would be advantageous to run long Fortran programs in the background while other tasks can be performed in the foreground.

FILE SYSTEMS

Time limitations have prevented all but the briefest discussion of file systems in general. The UNIX file system, which is essentially a tree structure, is particularly simple and interesting. Students are asked to study this system and to demonstrate their understanding of it by inserting and deleting files at various levels in the tree structure.

CONCLUSIONS

The concept of having a minicomputer readily available for hands-on use by students has been well received. This is especially true in view of the fact that while most students at Stevens have used the batch and time sharing facilities of the PDP10 system, very few of them have ever had the opportunity to actually operate a machine.

The process of starting out with a minimal paper tape system and ending up with a small but sophisticated operating system graphically displayed the need and value of an operating system. Furthermore, the various functions of the operating system took on added meaning when they were observed on an actual system.

The chief difficulty with the course was lack of time. The course met three hours per week for 14 weeks with the students being expected to perform all reading, written and programming assignments on their own time. Actually, this amount of time could easily be filled with a standard lecture course on operating systems without the availability of the minicomputer.

Curriculum revisions may allocate more time to the material in this course in the future. In the meantime, however, the following measures are being taken to make more efficient use of time:

1. Rely more heavily on suitable material in the textbook.[9]
2. Rely on the processor handbook[5] for the routine aspects of PAL11.
3. Utilize printed notes to summarize directions for the use of certain hardware and software facilities; e.g. the use of the text editor can be effectively explained via notes.

ACKNOWLEDGMENTS

The previously mentioned grant from the National Science Foundation and the financial assistance and encouragement of the Stevens administration are gratefully acknowledged. Particular thanks are due Dr. T. Gannon of Bell Telephone Laboratories and my assistant, Mr. J.S. Heilbronner for their work in implementing the UNIX operating system, and to Dr. O. C. Boelens of IBM and Professor A. C. Gilmore, Jr. of Stevens who were responsible for the original paper tape system. Also contributing to the success of this project were my colleagues, Professors P.R. Clement, R.C. Geldmacher, S. D. Shapiro and S. H. Smith.

REFERENCES

1. ACM Curriculum Committee in Computer Science, "Curriculum 68", *Communications of the Association for Computing Machinery,* vol. 11, no. 3, March 1968, p. 151.

2. Bell Telephone Laboratories, "Documents for Use With the UNIX Time Sharing System."

3. J. W. Cooper, *The Minicomputer in the Laboratory: With Examples Using the PDP-11.* New York: John Wiley and Sons. 1977.

4. J. J. Donovan, *Systems Programming.* New York: McGraw-Hill Book Co., 1972.

5. Digital Equipment Corporation, *PDP11/04/05/10/35/40/45 Processor Handbook.* Maynard, Mass., 1975.

6. R.H. Eckhouse, Jr., *Minicomputer Systems: Organization and Programming (PDP-11).* Englewood Cliffs, N.J.: Prentice-Hall Inc., 1975.

7. J. Lions, *A Commentary on the UNIX Operating System.* The University of New South Wales, Kensington, Australia, 1977.

8. J. Lions, *UNIX Operating System Source Code Level Six,* The University of New South Wales, Kensington, Australia, 1977.

9. S. E. Madnick and J. J. Donovan, *Operating Systems,* New York: McGraw-Hill Book Co., 1974.

10. D.M. Ritchie, *C Reference Manual,* Murray Hill, N.J.: Bell Telephone Laboratories, 1974.

11. D.M. Ritchie, *UNIX Assembler Reference Manual,* Murray Hill, N.J.: Bell Telephone Laboratories.

12. D.M. Ritchie, "The UNIX Time-Sharing System," *Communications of the Association for Computing Machinery,* vol. 17, no. 7, July 1974, pp. 365-375.

13. R. W. Southern, *PDP-11 Programming Fundamentals,* Ottawa, Ont., Canada: Algonquin College Bookstore, 1972.

14. H. S. Stone and D. P. Siewrorek, *Introduction to Computer Organization and Data Structures: PDP-11 Edition,* New York: McGraw-Hill Book Co., 1975.

15. K. Thompson and D. M. Ritchie, *UNIX Programmers' Manual Sixth Edition,* Murray Hill, N. J.: Bell Telephone Laboratories, 1975.

INTRODUCTORY PROGRAMMING COURSES WITHOUT LECTURES

George Nagy

University of Nebraska-Lincoln

Summary: Alternatives to lecture-based instruction are proposed for large enrollment introductory programming courses for non-computer-science majors. The instructional components to be substituted for the lecture sessions are described, with particular regard to the communications channels to be developed between the instructional staff and the students. The current courses are reviewed and a work plan is presented for implementing the proposal. Specific methods for administering and evaluating the experiment are advanced. Suggestions, advice, and reports on previous work in this area are solicited to lessen the risks associated with this large scale enterprise.

1. Objective

The objective of this paper is to solicit information rather than dispense it. At the University of Nebraska we are embarking on a rather radical overhaul of our introductory programming courses for non-computer-science majors. Since this endeavour involves significant faculty effort and a commitment by the University of about $150,000 per year, we are most anxious to avail ourselves of this opportunity to benefit from the collective expertise assembled at CCUC/9. It is our hope that by this means we will avoid many pitfalls and learn through the experience of others as well as our own.

Our intention is to replace the existing lecture-based method of teaching introductory programming courses for non-computer science majors by a more effective and less costly method based on more active student participation.

This proposal is motivated by the hypothesis that even in the current lecture setting, students of computer programming learn most of what they do learn from direct computer feedback in programming assignments, from problem-oriented readings in textbooks and programming manuals, and from personal, one-on-one interactions with each other, with teaching assistants, and with student programming consultants.

Student evaluations on file show that most students consider programming classes more difficult and far more time consuming than other classes. Furthermore, a very large fraction of the students - more than 50% in some classes - fail to complete these courses. There is, therefore, ample room for improvement.

The development of a more individualized teaching method not based on lectures in relatively small sections would also help the University keep up with the increasing enrollment in introductory programming courses. Many students have been turned away from such courses during the last two years because of the lack of available faculty.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

2. Proposed Method of Instruction

The proposed method will retain the current emphasis and dominant weight in grading on programming assignments, but will replace the lecture by alternative methods of providing students the information necessary to develop their programs.

At the beginning of the course, the students will buy a course syllabus, obtain computer accounts, and meet the faculty and staff connected with the course at the course headquarters. The prime candidate for course headquarters is the Remote/2 terminal room in Ferguson Hall, which currently houses 36 computer terminals, six keypunches, and a card reader/printer remote job entry station.

The syllabus will contain the learning objectives of the course, an explanation and rationale of the course organization, a schedule of topics, readings, and programming assignments, and extensive supplementary material specific to local circumstances and therefore not contained in the text and programming manuals.

Additional information, exercises, and quizzes will be dispensed to the students on demand through computer terminals and printouts at the time the class covers the material. Programming assignments and data for testing programs will also be "posted" through terminals and printouts. Students will likewise submit their solutions through the computer interface. This requires little additional overhead; when a student is satisfied that he has completed the assignment, or if he reaches the deadline for the assignment, he will simply route a copy of his program and output to the "instructor file" where it becomes available to the staff for grading and feedback. Solutions to the assignment, grades and instructor comments will be communicated to the students (on demand) through the computer.

Instructors and teaching assistants will be available twelve to sixteen hours a day to provide help at course headquarters and through telephoned requests. A HELP file for queries and instructor replies with at least daily turnaround will be provided for students who prefer to work at other computer locations or at unusual hours.

Each student's progress will be monitored by the instructors on the basis of assignments submitted, quiz scores, number of programming runs, time spent logged on at the terminals, and HELP file activity. Students in apparent trouble will be requested to schedule personal interviews to obtain additional assistance.

The following simple additional features are to be implemented to improve all-around communciations in the class:

a) A suggestion box file for facilitating student input on course related matters.

b) A student grades summary statistics file which will show students their progress in the course relative to others (students, cannot, of course, access each other's grades.)

c) An important notices file to inform students about changes or malfunctions in the computer system, changed assignment deadlines, newly discovered errors in the text or other material, films, video broadcasts, or interesting campus functions related to the course.

d) A correspondence file to allow notes and queries to be directed to and received from other students.

e) A calendar to facilitate scheduling personal interviews with the faculty and staff.

3. Current Courses

While the largest number of introductory programming courses are offered in the Computer Science Department (14 sections per semester, not including the courses for majors), such courses are also offered in the Colleges of Engineering and Technology, Architecture, Business Administration, and in Teachers College.

The following table lists the courses that have been identified:

CS 110	Introduction to Data Processing
CS 150	Introduction to FORTRAN Programming
CS 237	FORTRAN Programming for Business
CS 238	COBOL Programming for Business
CS 300	Introduction to Computing in the Social Sciences
EM 112	Methods of Communication in Engineering II
MGT 337	Electronic Data Processing for Business
ARCH 218	Computer Applications in Architecture I
ED 498	Computers in Education Workshop (summer only)

The total enrollment in these courses is well over 2,000 students per year. The Colleges of Engineering and Technology and of Business Administration and some departments in Arts and Sciences require each student to take a programming course, while elsewhere such a course serves as an elective to fulfill science or mathematics requirements. Some graduate students take it as a research tool, others out of thesis-related needs. Others enroll to acquire the rudiments of a skill which they expect to prove useful in subsequent courses or their professional career, or merely to familiarize themselves with an all-pervasive aspect of modern technology.

All of these courses are taught by means of lectures, either in small sections (20-40 students) without separate laboratory or recitation, or in large sections (up to 150 students) with smaller lab sections staffed by graduate teaching assistants. In each course the students are expected to complete from 5 to 10 programming assignments of varying degrees of difficulty, requiring a total of 20 to 100 computer runs. The students use either the IBM 360/65 or 370/148 computers at the University, or the IBM 370/158 at the State Capitol. Access is through punched cards or remote terminals. The computer allocation per student varies from about $50 to $130. Ninety percent of the students learn the FORTRAN language, ten percent the COBOL language.

It is expected that the computer configuration available for teaching such courses will remain roughly the same for several years. In the meantime, the continuing increase in enrollment and the lack of foreseeable growth in the faculty qualified to teach computer courses will lead to a gradual increase in the size of the sections and a concomitant deterioration in the learning experience of the students.

Several hundred textbooks are currently available for introductory courses in either FORTRAN or COBOL. There has been no attempt to coordinate textbooks, course content or teaching methodology, but some standardization is effectively imposed by the specific language processors (compilers) available at UNL. To a large extent, the mechanics of each course are determined by the input/output facilities located near each department. For example, most engineering students use punched cards in Nebraska Hall, business students use typewriter terminals in CBA, and students in computer science use display terminals in Ferguson Hall. In these respects, the situation at UNL is not markedly different from that at any other university surprised by the rapid and disorderly intrusion of computers in the curricula.

4. Work Plan

The following tasks have been identified as necessary to accomplish the stated objective. They are not listed in the actual order of execution since several may be worked on concurrently.

a) Preparation of detailed course and learning objectives. These are expected to be much more detailed than those on record for the current courses.

b) Collection of information on other similar programs. Literature survey and telephone investigation, perhaps visits, to discover and study other self-paced large-enrollment introductory programming courses at other institutions. This presentation is part of our efforts to discover new leads.

c) Textbook selection. Since the preparation of an entire text on a programming language is beyond the scope of the project, existing texts and manuals must be carefully reviewed with regard to teaching strategies and all programming examples must be adapted to the locally available systems.

d) Preparation of ancillary material necessary for the operation of the local systems. Log-on procedures, remote operating systems, exec procedures, and job control language vary enormously from installation to installation, and cannot therefore be expected to be adequately covered in the selected texts and manuals. Much of this material is already available from existing courses such as the current hands-on Orientation course (NUROS modules.) However, better documentation on compilers and on several locally implemented programming packages need to be made available at easily accessible locations on campus.

e) Development of examples suitable for students with various backgrounds. Engineering students are most comfortable with programming problems drawn from engineering, business students with data processing or modeling examples, social science students with examples from the behavioral sciences, and so on. Examples for each group must illustrate the same programming concept and be of equal difficulty, yet they must be couched in the setting of the appropriate discipline.

f) Survey of audiovisual aids. Some general concepts relating to computers, including software and hardware configurations and programming style, may be best illustrated in film strips, motion pictures, or video tapes, hence candidate material in these areas should be screened. The Computer Science Department maintains a file on such materials.

g) Preparation of functional computer programs for distribution to students at the beginning of the class. Such programs can be successfully executed by all students and serve to avoid a frustrating introduction to the computer. Students can then gradually learn to modify their programs, beginning with only one or two statements or data items at a time. In this manner the students may derive positive feedback from the computer at the important early stages where many students currently simply give up.

h) Definition of course logistics. As in any large enrollment laboratory course, the course administration must be carefully planned in advance. The distribution of computer accounts, the scheduling of teaching assistants at the computer terminal locations, the training and supervision of teaching assistant, the scheduling of interviews with the faculty in charge of the course, the tours of the Computing Center, the balancing of the load on the computer and on the physical facilities, and the grading scheme, all require much thought and planning.

i) Development of on-line computer material. Since the students in the proposed course must frequently interact with the computer, much of the information relating to both the conduct and the contents of the course will be presented to them through this interface. Such items as programming assignments, quizzes, grade standings, consulting schedules, and explanations of specific difficulties can be readily provided on demand at terminals and on printouts. Conversely, students should be able to submit the results of their work by computer, although we do not expect that the evaluation function itself can be wholly automated in the near future.

j) Implementation of a message system which will allow students to communicate with each other and with the course instructor regardless of possible schedule conflicts. Such systems are already available at most computer facilities, and can be readily implemented on the UN systems.

5. Evaluation

We are still soliciting ideas on the difficult matter of evaluating the proposed method. Here are some ideas we have collected so far.

A pilot section of the course will be offered in the fall semester of 1978-79. Students will be randomly assigned to either the traditional lecture method or to the pilot course in order to provide the basis for a controlled experiment. The evaluation itself, or the measures of effectiveness that will be used to compare the two groups, will include measures of cognitive tasks as well as measures of affective and behavioral responses to the two teaching situations. The measures that seem most useful will continue to be administered even once the course is adopted in order to monitor the quality of instruction and to test further improvements.

Cognitive measures:

a) Identical quizzes and programming assignments will be given to both groups of students. These will be designed to reflect the competencies defined as the course and learning objectives.

b) The observations of teachers of courses requiring an introductory programming course as a prerequisite will be solicited to determine their satisfaction with the preparation of the students. There is, inevitably, a delay of at least one semester in this form of evaluation.

Affective measures:

c) Student evaluations during and at the conclusion of the course, and two years later. Since many of the prospective students are freshmen, we would expect to be able to reach them as juniors to see whether they have made any use of computers in the meantime and to solicit their opinion of the course.

d) In-depth open-ended and structured interviews with given students, conducted by persons not associated with the project. Some of the ADAPT experience may be useful here, and the help of the Teaching and Learning Center may also be solicited.

e) Evaluation by the faculty connected with the course.

Other measures:

f) The course drop rate for both groups.

g) Polls of responses to any questions relating to issues as they occur during the project. Such responses can be easily collected, tabulated, and analyzed as part of the students' regular interaction with the computer.

h) Tracking the complete student-computer dialog of selected students to pinpoint specific difficulties with either the course mechanics or with the subject matter. This information can be used to improve the course delivery in subsequent offerings.

i) Provide a selected sample of students (or, for that matter, all graduates of the course) with limited computer accounts and see what they do with them, if anything.

6. Improvement in the Quality of Instruction

Our expectation that the proposed method of teaching introductory programming courses will improve the quality of instruction is based on the following considerations.

a) Greater emphasis will be given to the mode that seems most effective in student learning, namely immediate individual feedback (in person, through the telephone, from computer error messages, through the HELP and correspondence files, from both instructors and other students).

b) Provision of exercises and problems drawn from a wide variety of disciplines and suitable for students with differing backgrounds. Currently only the largest constituencies can be served with customized courses, and additional limitations are imposed by the scheduling of lecture sections and of instructors with the appropriate preparation.

c) Greater resources--due to pooled effort--will be available for keeping the information provided to the students up to date with the constant evolution of the hardware and software configurations of the three major computing systems serving the university.

d) The use of the computer for submitting and returning assignments and for monitoring students will render it possible to give shorter and more frequent assignments and to call attention earlier to students in trouble.

e) The additional computer interaction required of the students, though relatively simple, has its own pedagogic value as a striking demonstration of the file-management and communications capabilities of computers. In a sense, the "dehumanization" of instruction may appear as an asset in the current context rather than as a potential liability as in social science or the humanities.

f) The increased staff time resulting from the elimination of lectures and the reduction of paper work may lead to more individual interaction with the students and improved coordination among the staff. Currently, for instance, it is almost impossible to schedule a conference among the instructors responsible for the sixteen lecture sections of the introductory computer courses for business students.

g) Improved consistency. Many sections of the programming courses, particularly those offered under the auspices of Continuing Education, are currently taught by part-time staff. The proposed system offers the possibility of increased standardization of the course offerings on a semester-to-semester basis, providing a more stable basis for subsequent courses which require a knowledge of programming. The present distinctions between day and night classes would be largely eliminated, and the lack of a binding time-table would allow increased participation by non-traditional students.

h) Counting transit time, students will find themselves with about six hours of additional time per week, which will serve well those who devote it to course work, either in this class or in others.

The above arguments contrast the proposed course with the current lecture-based organization. We should, however, mention that we have also considered both a Computer Aided Instruction approach and a strict Keller Plan approach.

We rejected CAI for the time being on grounds that neither the hardware nor the software to support large-scale CAI are currently available at UNL, and that the effort required to develop acceptable courseware for the local systems is beyond our capabilities. We intend, however, to continue monitoring closely courseware developed elsewhere, and regard the current proposal as a step in the direction of CAI.

We have adopted some ideas from PSI, but decided to enforce a common timetable for the class. Because success in programming is fairly closely correlated with the amount of time students devote to it, it is almost impossible to catch up once a student lags behind. The common timetable will also simplify the class organization and increase opportunities for peer communication.

7. Project Administration

The project team is to consist of the director, two other faculty members, and two graduate students. The participants will meet on a weekly basis to partition and carry out the tasks listed in Section 5. An advisory board (3 faculty members from the Computer Science department, 4 faculty from outside the department, 4 non-computer science students, and two unpaid, practicing data processing professionals) will provide additional ideas and meet six to nine times per year to monitor and critique the progress of the project.

Sufficient preparation for a pilot offering of the course will be conducted during summer 1978. In the first semester of 1978-79 an experimental version of the course for about 50 students will be offered with the help of the faculty and graduate assistants who participated in the planning. In the spring semester the results of the pilot experiment will be analyzed as decribed in Section 7 and a decision made whether to proceed further. In the event of an affirmative decision, curriculum committee applications will be filed to consolidate the current courses under one number, with the emphasis on different disciplines accomodated in specific sections. During the summer of 1979 additional course materials will be prepared, training sessions established for teaching assistants, and provisions made for sustained evaluation. The course will be offered on a regular basis from fall 1979-80.

Whether the experiment is successful or not, it will be reported at CCUC/11, and an attempt will be made to publish the results in the journals of the appropriate professional societies and in the broader circulation education-oriented periodicals.

Acknowledgments

This plan was developed collectively by the entire Computer Science Department, and contains major contributions from Professors David Embley, Dennis Leinbaugh, and George Corliss. The author also had the benefit of constructive suggestions from Professors Gerald Boardman and Virginia Corgan from Teachers College, Gerald Smith from Engineering and Technology, Richard Schonberger and Margret Hazen from the College of Business Administration, Donald Costello from the University of Nebraska Computing Network, and Delivee Wright from the Teaching and Learning Center.

INTRODUCTION TO COMPUTING - A MODULAR APPROACH

Phyllis A. Kent and Marguerite A. Boisvert

Syracuse University Computing Center
Syracuse, New York 13210
(315) 423-2677

At Syracuse University the needs of departments, colleges and individual students for introductory programming courses vary considerably. Each year the School of Management must introduce approximately 500 students to COBOL, computer concepts and business applications; mathematics, engineering and chemistry would like to require a solid background in APL for their majors; other departments favor FORTRAN; and the School of Computer and Information Science would like their computer science majors to complete a series of one credit courses in each of several programming languages.

The full spectrum of needs covers three levels of introductory course work and a number of different computer languages. The lowest level is an introduction to problem analysis, computer concepts and terminology. This is followed by an introduction to a programming language, and finally the third level is a course that provides extensive programming experience in one language. In an attempt to satisfy all of these requirements we have developed a sequence of variable credit modules that may be taken in a variety of combinations.

Course Descriptions

The course modules provide the equivalent of a three credit course for each of the languages APL, FORTRAN and COBOL. The same introductory material is taught in the first third of all three sequences. The second third of these courses introduces a programming language and the last third is project oriented. APL and FORTRAN students complete a programming project; COBOL students study computer applications in business and write a paper on one application.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

Introductory Module

The material covered in the first third of all three sequences includes an introduction to problem solving and to the idea of an algorithm. Flowcharts are discussed early in the course and homework assignments as well as classroom work lean heavily on the flowchart as a tool in problem solving. Concepts such as initializing and incrementing counters, looping, testing and branching are illustrated with simple problems. The history of computing, information on hardware, software, data structures, etc. are introduced using commercially produced video tapes.

At the end of this module a two part examination is given. The first part consists of multiple choice questions on the video tape materials, and the second part requires that the student flowchart an appropriate solution to a fairly complex problem. Students who pass the examination receive one credit for completing this module. They may then elect to continue with an introduction to a programming language.

Language Modules

Each language module is taught as a variable credit course. Students who register for a language module may complete one or two credits. To receive the first credit, the student must demonstrate a basic knowledge of the syntax of the language taught, and must be able to use that language to express problem solutions similar to those discussed and flowcharted in the introductory module.

All homework assignments require the use of the computer. Assigned programs are designed to illustrate the language characteristics as they are introduced in the classroom and, as the course progresses, the assignments become logically more demanding. An examination at the end of the language study requires that the student interpret the meaning of program segments and write program segments to solve specific problems.

After successfully completing the one credit study of language fundamentals, the student may elect to continue with the second credit of this variable credit module. Students choose a project from a set of projects approved by the instructor. Typical projects are: a simple airline reservation system, a banking system, or a game.

Before attempting any programming, a student must hand in a short description of the proposed project and a rough flowchart that outlines the logic of the problem solution. Programming may begin as soon as the instructor has approved of the proposal. While students are working on their projects, no classes are held. Instead, students may individually meet with the instructor by appointment. The purpose of these meetings is to give the student whatever help and/or guidance is needed. To complete the one credit for the project, a student must hand in listings of all programs, sample runs, flowcharts and written documentation.

Course Management

Careful organization and management of a set of courses with the complex structure described above is one of the keys to success of the plan. It would be less important, perhaps, if only a few students were involved, but with a potential enrollment of 600 students per semester, course coordination is essential.

The programming courses were given an academic home in the College of Arts and Sciences but with non-departmental status. This decision was based on a number of considerations. Since non-science students are often intimidated by the common association of computing with mathematics and science, the non-departmental designation made it possible to avoid this association. Non-departmental status also provided for an all-college faculty committee to approve the syllabus and instruction.

In the past many academic departments expected students to acquire some computing skills for use in course work and to a great extent this need was filled through non-credit mini courses taught by the Academic Computing Center staff. However, students have been vocal about their desire for an academic record of their computing achievement. Because faculty in academic departments cannot easily assume the additional burden of presenting introductory programming to their students and because the Computing Center has been providing this service in an informal way, the Academic Services staff of the Computing Center was asked to assume responsibility for course management. Courses were designed by Computing Center staff and were approved by the College of Arts and Sciences and the University Senate.

There is an historical precedent for the delegation of this management and supervision to the Computing Center staff. The Center has a long history of offering high quality non-credit courses in introductory computing which are very popular among students and faculty. Committees have recommended that these courses be approved as credit courses, but the new course sequence makes this unnecessary. The Center is uniquely qualified in many other ways to provide this course supervision. It employs a number of well qualified computer professionals who have experience in teaching at all levels, and who are in the mainstream of changing computer technology. The Center has the facility to produce and distribute, at minimal cost, large amounts of technical documentation. Packets of materials can be prepared for different course modules as needed each semester. These packets will reflect the current state of hardware, software, and user procedures, and therefore, will help to avoid the frustrations students feel when they are given incorrect or out-of-date material.

The large number of students involved, and the variable-credit, continuous-registration nature of the courses present some serious logistic problems. Among these are:

1. Scheduling class assignments in such a way that computer resources are not overburdened. Keypunch and remote terminal needs can be scheduled to avoid peak loads.

2. Computing Center AID staff, which normally assists computer users, will have copies of all assignments, and will be instructed as to the amount of assistance they may give.

3. Video tape sessions are common to all courses, and must be scheduled for viewing and reviewing at convenient times in the Computing Center Audio Visual lab as well as other sites.

4. Individual computer accounts for funds as required by the course modules must be issued to all students and carefully monitored.

5. As each module is completed, registration procedures for those students eligible for the next module must be implemented.

On a less formal basis the services described here are among those already provided for the entire University community by the Academic Computing Center. We believe that this extension of Academic Computing service can be carried out in a cost effective way that will benefit both the Academic departments and the students.

Experience to Date

The original versions of these courses were standard three credit offerings. Student response to the courses in this format was very good. In general the programming projects were of high quality and the final reports were excellent. However, all the instructors agreed that the project was more than some students could effectively manage, and this realization led to the decision to make the project an optional third credit. On the other hand a number of students liked this approach to programming so much that they wanted to repeat the sequence for a different language. This resulted in the restructuring that created a separate course for introductory concepts.

As the course is now structured considerable flexibility is allowed. Students need not repeat material already familiar to them when they choose to study a second or third language. In addition, by electing one credit of work from each language module, a student can acquire the fundamentals of three different languages with a minimum investment of time. Further, those students who desire a more thorough grounding in programming, may acquire considerable real experience from their work in designing, debugging, and documenting a programming project.

Our decision to eliminate classes and work with students on a one-to-one basis during the project phase was also very successful. Students felt that the opportunity to discuss problems with the instructor and receive guidance throughout the project boosted their confidence and provided an important learning experience. Recently we have received a number of inquiries from students who would like to expand on their projects and receive additional credit. Therefore we are considering allowing a fourth project credit for interested students who have demonstrated the ability to work independently.

Summary and Future Plans

Experience with introductory programming courses has suggested changes in both content and format, and many of these changes have been implemented. We are presently considering the provision of an alternative to the programming project for some students. While students agree that the project is the most rewarding part of the course, some students are simply not skilled enough for the independent work required by the project. An alternative might be to extend class lectures and required assignments for those students who do not reach a sufficient level of achievement by the end of the language credit and let the project be a reward for high achievement.

A few other considerations are: A clear need exists for improved audio visual materials on computer hardware and software and for inexpensive student work books; we have found no appropriate text for introductory material on flowcharting and problem solving, although parts of large expensive texts would be useful; audio tapes on language units would be valuable as a review.

As this paper goes to press, a number of departments of the College of Arts and Sciences, the School of Management, the School of Computer and Information Science, and the College of Engineering are considering making some modules of this course sequence a requirement for their undergraduates.

COMPUTER AIDED INSTRUCTION IN BUSINESS STATISTICS

Chiou-Nan Yeh
John A. Willhardt
Alabama State University

INTRODUCTION

During the last two decades, business use of computing techniques has increased immeasurably. Neither large nor small businesses can remain competitive without taking advantage of the computer's speed, accuracy, and ability to process large volumes of information. Operations Research techniques, the application of scientific methods of analysis to executive-type problems, and the processing of accounting and financial information have been applied to establish control of the operations with optimum solutions to problems. (2:5) Statistical sampling allows fairly rigorous conclusions to be drawn in many cases concerning needs or problems. (1:369)

Although the computer is a vital part of business and society today, very few undergraduate courses in statistics include knowledge and use of computing techniques. Many statistical methods and formulae are too tedious to handle without the use of the computer. According to Schneider, computer supported instruction in quantitative methods classes can produce a positive attitudinal change regarding computers and their value. An important goal of education in business administration is to increase student awareness of the value of computers in business and industry. (3:327)

Many of the analytical methods introduced in the statistics textbooks call for an unpleasant amount of arithmetic, particularily, when the sample size is large. The examples used in the majority of texts have been tailored to keep down the amount of mathematical work, but the business world is not so kind. Advances in computer technology have eliminated much of the tedious and routine work; the user needs only a slight knowledge of the computer to get the answer or information needed. It is not necessary to understand how the computer works or how to program it. Libraries of frequently used programs are stored for use as needed.

This paper presents some experiences and results on the efficiency of a teaching method used in Business Statistics during the Spring and Summer Quarters of 1977.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

317

MOTIVATION AND OBJECTIVES

In an effort to introduce statistical computer techniques to business students, four faculty members in the College of Business Administration at Alabama State University developed a workbook to teach undergraduate statistics with the aid of the computer. This workbook is designed for the beginning business student to introduce the student to the concept of statistical analysis and provide the student with an insight into and appreciation for using the computer to extract statistical information for business decision-making. The pedagogical approach of the workbook is built around the twin ideas that the computer can free the student from laborious computation and that it makes possible a greater understanding of statistical principles through creative experimentation.

The workbook consists of eight units. Each unit includes the description of a statistical concept or method, followed by a computer program and detailed instructions for using it. The sets of computer programs are such that their use requires no prior computer knowledge. The only skill a student has to possess is the ability to use the card punch.

In the process of its experimental stage, this teaching method in business statistics is expected to achieve the following objectives: 1) to increase student achievement, as measured by course examination scores and by the comprehensive test scores which each business student is required to take before graduation; 2) to attract more majors in the computer and quantitative area. Alabama State University has an open admission policy in regard to recruiting students. As a consequence, there have been tremendous variations in students' needs and abilities. Most students lack adequate background for statistics courses. They have difficulties in handling simple arithmetic; thus, they try every possible way to avoid taking courses in the quantitative area. Students, uninitiated in computer usage or exposure, exhibit anxiety when confronted with the computer for the first time. It is the authors' belief that a pleasant experience with the computer will serve to reduce this anxiety. The ease in using the computer programs should provide a pleasant encounter with computing methods. Hopefully, this experience will arouse the students' curiosity and creativity in further computer utilization.

PILOT PROJECT AND ITS PRIMARY RESULTS

A pilot project using computer programs in business statistics courses was implemented during the spring and summer quarters of 1977. The primary objective of the project was to provide the students with enough exposure to the system in order to obtain a preliminary objective appraisal. Additionally, classroom experience with computer programs was needed in order to identify desirable modification.

After a statistical tool or method was taught, the students were given two handouts, the first one contained the required job control cards to obtain the relevant computer program from the computer library (these programs were put on the core image library or a private library), and the second handout contained the data (homework assignment). The students were required to punch both the job control cards and all of the data. The first handout contained the required order of the cards. When the student completed the punching and put the cards in the proper order, he turned them in at the computer center and picked up the printed results later.

The following statistical programs were written for use in statistics classes: 1) Frequency Distribution, 2) Averages and Weighted Average, 3) The Median and The Mode, 4) Variance, Standard Deviation, Coefficient of Variation, Moments, Skewness, and Kurtosis, 5) Relatives, 6) Method of Least Squares, 7) Regression Analysis, and 8) Proportion Estimation.

For statistics classes, the instructor selects those programs he wishes to use. The programs are processed using an IBM 370 model 115 Computer with 196K.

Generally, with the first program, most students needed two or three computer runs as any error in punching or card order prohibits the program from processing correctly. Therefore, at the beginning of each quarter, a computer lab assistant or the instructor must be available to help the students.

After implementing this project for two quarters, the authors recognized several advantages of the computer-aided instruction in these courses: 1) Students are able to work more difficult statistical problems in much less time. The computer exercises allow the student to concentrate on the data interpretation and statistical experimentation rather than tedious number-crunching. 2) Student achievement was positively and significantly related to the number of homework assignments completed, and the percentage of classes attended. The number of students who submitted homework by the due date was larger than before the computer programs were used. The majority of the students in the statistics classes attended all the sessions. As a consequence, the student achievement was enhanced as measured by the test scores. 3) The knowledge gained from utilizing computer equipment can be applied in other disciplines. 4) Students learn how to use computer programs in handling huge volumes of data.

When implementing this project, the instructor should be aware of the class time constraint. Additional class time was required for instruction in the use of the card punch.

STUDENT EVALUATION OF THE PROJECT

At the end of the summer quarter of 1977, questionnaires were given to the students in the Business Statistics classes. These questionnaires and their corresponding responses are listed as follows:

1. The use of a computer in statistics classes makes it possible to use more real-world like homework exercises.

a) Disagree	3.57%
b) Agree	67.86%
c) Strongly agree	28.57%

2. How much help do you think a computer can be to the student to aid his understanding of the material in statistics classes?

a) Little help	0.00%
b) Some help	32.14%
c) Much help	67.86%

3. Computers are difficult for a person to learn to utilize.

a) Strongly disagree	10.71%
b) Disagree	71.43%
c) Agree	17.86%

4. At least one computer programming, data processing, or MIS course should be a part of the curriculum for undergraduate business majors.

a) Disagree	3.57%
b) Agree	50.00%
c) Strongly agree	46.45%

5. It is important for a businessman of today to be able to utilize a computer.

a) Disagree	3.57%
b) Agree	42.86%
c) Strongly agree	53.57%

6. A computer can provide better information to management.

a) Disagree	7.14%
b) Agree	60.71%
c) Strongly agree	32.14%

7. The College of Business Administration should offer a course in Computer Applied Statistics.

a) Disagree	17.86%
b) Agree	64.28%
c) Strongly agree	17.86%

8. Would you take a Computer Applied Statistics course if you had time and opportunity in your academic program?

a) NO	17.86%
b) YES	82.14%

Based on the responses to the above questionnaire, the authors concluded that students responded favorably to the use of computer programs in Business Statistics courses. They had no significant hostility to the utilization of data processing equipment. Moreover, students urged that the computer programs should be used in the future statistics classes and in other disciplines as well. The student's evaluation of this project was largely unrelated to his/her classification, major, or academic standing.

CONCLUDING REMARKS

The initial results of the pilot project at Alabama State University indicate that computer programs can be utilized effectively in Business Statistics classes.

Viewing the professor in a dual role as a researcher and teacher, the implementation of the computer programs could enhance the two roles and enable a professor to fulfill both functions simultaneously.

The research culminating in the Workbook was supported by the Alabama Center for Higher Education in Birmingham, Alabama.

BIBLIOGRAPHY

1. Murdick, Robert G., and Joel E. Ross. Information Systems for Modern Management. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1975.

2. Stockton, John R. and Charles T. Clark. Introduction to Business and Economic Statistics. (5th ed). Cincinnati: South-Western Publishing Co., 1975.

3. Schneider, Kenneth C. "Computer Supported Instruction in Statistics: Long Range Implications." Decision Sciences. VIII, 1 (January, 1977) pp. 318-331.

THE COMPUTER IN AN INTRODUCTORY STATISTICS COURSE

K. Rappaport
T. Stafford
Essex County College
Newark, NJ 07102

The computer is becoming increasingly more important to almost every academic discipline. In addition, it is an integral part of business, government, and even everyday affairs. Anyone who trains in statistics, who will handle data, or work with people who handle data must have a certain minimum competency in the use of the computer. "The purpose of introducing and utilizing computer-related ideas is basically the psychological one of getting the student involved with and thinking about mathematics... The Panel has more than adequate anecdotal evidence that the injection of computing will provide very strong motivation for the student. A second reason is the ubiquity of the computer in our present society and the need for the students to understand its potentialities and limitations in order to function as citizens and employees."[1]

As participants in the Second Conference on Educational Computing in Minority Institutions (ECMI/2) sponsored by NSF in March 1976, we became aware of the many possibilities for integrating the computer into the course curriculum. Upon returning to our institution, Essex County College, Newark, New Jersey, we sought the opportunity to integrate the computer into our statistics curriculum. Although our major concern was with the instructional use of the computer, it was necessary to become involved in other aspects; phones, space, and equipment had to be found; the

---

[1] "A Course in Basic Mathematics for College" Committee On Undergraduate Program in Mathematics, p. 17, 1971.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

program had to be justified and fit into the budget; programs suitable for student use had to be found or written. Since hard copy terminals were planned for individual student use, programs were needed that would be conversational, easy to use, and helpful in the learning process.

Computer facilities at Essex County College are not easily available for academic use. However, through community involvement and much legwork, the Mathematics Department obtained a grant of computer time and some equipment from outside sources. The College has budgeted space for a terminal room, as well as money for phones and equipment.

The computer system contributed for our use was a PDP 11 family capable of running an RSTS operating system. The programming language used for this system was BASIC PLUS. Both the computer and programming resources were made available to us by Prudential Insurance Company. They have indicated a willingness to make the statistics program available for educational purposes.

Since September 1976, the computer has been used experimentally in five sections of statistics being offered to Essex County College students. A similar type of statistics course was also developed for gifted students attending local Newark high schools. This course was run initially in January 1977 and continued in subsequent semesters. The high school students have as their math background a minimum of two years of high school algebra.

It was hoped that the use of the computer would act as an educational motivator enabling students to see the overall use of statistics while limiting the computational drudgery. Our other aim was to go beyond the traditional subject matter covered in an elementary statistics course and to acquaint the students with realistic statistical applications. The course endeavored to develop the techniques necessary to make statistical analyses of large amounts of data and to give students "hands on" experience in using the computer to solve statistical problems.

A course outline with computer applications was developed so that it could be used with any standard statistics text. Every chapter had a corresponding computer assignment. For example, the introductory chapter had as a requirement the development of competence in the use of the computer to do summations. All topics were first assigned as in a regular class; the problems could then be checked using the computer or similar problems with larger groups of data could be solved using the computer. The section on measures of central tendency included first finding the mean, median and mode using the appropriate formulae for both grouped and ungrouped data. The computer was then used to find these same measures for student collected files. It was hoped that by looking at this information, the student would get a "feel" for what the mean, median and mode show.

The text was not computer oriented so handouts for the computer sections were written. These handouts (see Appendix I and II) consist of samples of the computer printouts and give an example of the interactive nature of the programs.

In the first semester that the course was offered, no class time was given to computer use. Students were required to complete assignments by using the terminal room during any of its scheduled hours. The terminal room was available for student use a total of fourteen (14) hours per week. The hours were spaced throughout the week, the earliest being 9:30 A.M. and the latest 7:00 P.M. and an instructor was present at all times. For better results, it was felt that computer lab time should be scheduled as part of the class. Consequently, for the September 1977 semester, the students were scheduled into the terminal room a total of two hours per week. During this time, an instructor or tutor is available to help. The students are now forced to use the terminals early and often in the semester and are encouraged to learn each program separately rather than all at once.

In the original course offering, no specific assignments, aside from a project, were made on the computer and no time limits were given. Students had the semester in which to familiarize themselves with the basic programs which would be needed for the project. It became apparent that more structure was needed so specific assignments and timed computer tests were scheduled. A series of quizzes, asking the students to call various programs and analyze certain files, were given. Each student was now required to use the programs as they occurred during the semester.

The students were introduced to computer terminal usage by assigning each student a string of data to be entered. The data was taken from the students themselves. It included: height, weight, age, pulse, number of cigarettes smoked per day, and rank in the family. In this way, each class created its own file to be used later for statistical computations. The students were asked to determine various conclusions from the above data. After collecting the information from the computer, they were asked to organize the data in such a way that their particular questions were easily answered.

The computer was thus utilized in two ways, as a computational tool to check or review homework and to do actual statistics on class data. Handouts representing these two aspects are attached. (See Appendix I, II) It can be noted from these handouts that the programs made available for use were semi-conversational. The confidence interval instructions used the computer for computation only. It required the student to prepare given values before using the program; critical scores had to be looked up on tables and standard deviations had to be calculated. In fact, this type of problem could have been done as easily on a calculator.

The instructions for elementary statistics represent the use of the computer for the purpose of analysis. The file used (ROSES) contains data from a STAT II class. This program finds the maximum, minimum, mean, standard deviation, standard error and range for all the variables, in this case six (6). It is helpful to compare these values with the printout of the data and provides an opportunity to enhance the concepts while removing the drudgery. After having done the calculations by hand for the different variables, the students had a great deal of appreciation for the speed and accuracy the computer can provide.

Another way the computer was used was to enhance instruction. It made possible repeated printings of histograms for the same data with different class intervals. This helped clarify the effect of class size on the histogram and showed how the data could be presented differently. At the same time, the students were able to see how the histogram demonstrates measures of central tendency. (See Appendix III for sample printout)

A class project covering the semesters work and utilizing the computer was assigned. In this manner, the students were involved in data collection, organization and development of hypothesis and variables, use of the computer, analysis of data and obtaining results, thus enhancing their appreciation of the entire procedures involved in statistical analysis. Each student collected a sample, which was entered into a class file; statistics were obtained for the file and either questions were answered or a hypothesis was tested. The particular project attached was a consumer price study of a market basket. Readings were assigned on the topic and various items were selected by the class for study. Depending on whether it was a STAT I or a STAT II class, different questions or hypothesis were tested.

In the STAT II classes, the students were asked to test a hypothesis and find confidence intervals. Before using the computer, it was necessary for the student to know the critical score, the sample size, the mean and standard deviation of their class file. In order to find these last two pieces of information, the student had to call the elementary statistics program and analyze the program to find information on the printout pertinent to their individual project. The student then had to collate this information in order to use it in the CALCT program to find the confidence interval or the "z" score for the hypothesis testing procedure. The actual hypothesis test had to be done by the students using the data they obtained from the computer.

After testing the hypothesis, a student could then look at the data on the printout and try to determine how his results compared with the actual array of data. It was interesting to note student reactions to the trends apparent in the data. Student interest was generated in discussing the overall picture presented by the data since they had first-hand knowledge of what these numbers meant and where they came from.

As students entered data, it became apparent that those with little or no typing experience had difficulty adjusting to the keyboard. This resulted in reluctance on the part of some students to use the terminals. It also forced others to spend long periods of time sitting at the keyboard looking for letters or numbers, causing a backup in the terminal room during peak usage hours. This problem was reduced with the increase in the number of available terminals. We started with three hard copy terminals. That has been increased to seven as of January 1977. Our equipment consists of one Hazeltine 2000 and six Decwriter LA 50's.

Student evaluations indicated that the computer applications had made the statistics more relevant and in many instances had been useful to them in their work. In general, collection and organization of data gave the students a better idea of the statistical process; the presentation of large amounts of data on the computer enabled students to see trends and make intuitional conclusions about the data. The numbers took on meaning in the applications which is lacking in a standard class. The involvement in the statistical process and computer use not only sparked student interest but also enabled them to overcome fears and biases about computers. At the same time, it opened up more possibilities for their future careers.

The availability of the computer made a large class project a worthwhile exercise, not just busy work. Large amounts of data could be analyzed without the tediousness of hand calculation. This process also emphasized the interaction of computer and researcher. The need to analyze the computer printout before proceeding in the steps of a problem showed the importance of the individual in the computer application. In future classes, this aspect will be expanded with the addition of smaller projects. To emphasize computer use as a skill, specific goals need to be set, as well as periodic evaluations of student facility with the different programs. In this way, both computer use and statistics can be learned and their combination be a fruitful educational experience.

APPENDIX I

COMPUTER INSTRUCTION FOR CONFIDENCE INTERVALS

The computer can construct a confidence interval for any problem using a normal distribution if you give it the proper information, including the critical Z score. The procedure is the same as the one you used for hypothesis testing.

Question	Your Response
READY?	OLD CALCT
READY?	RUN

After finding the program, the computer will ask you for more information.

Question	Your Response
Select Routine by Number from the following list. List (Y, N)?	You type Y if you need a copy of the list. Type N if you have a copy. (You should have one from the hypothesis testing unit)
Routine Number (or 999)?	Pick the number which corresponds to your problem under confidence intervals. As an example select 5.
FORMAT: SAMP MEAN, CRITICAL (Z) SCORE, STD. DEVIATION, SAMPLE SIZE?	Type in the four values separated by a comma on one line. In the example use 100, 1.96, 24, 36.
CONFIDENCE RANGE: 92.2 TO 107.8	This is the confidence interval.

Question	Your Response
Routine Number (or 999)?	If you wish to continue select another number. If you are finished, type 999.

APPENDIX II

ELEMENTARY STATISTICS INSTRUCTIONS

This program is used to find the maximum, minimum, mean, standard deviation, standard error of the mean and the range.

Follow the same procedures as before to start the program.

General Starting Procedure

Question	Your Response
READY?	OLD STAT 12
READY?	RUN

Once you have the program started, the following is the procedure.

Question	Your Response
FILE NAME?	Choose one of the files, i.e., ROSES
ENTER ANALYSIS TO BE PERFORMED	ELEMENTARY STATISTICS
TYPE NUMBER OF ROWS & COLUMNS	This will depend on your choice of file for ROSES, it is 22, 6
ROWS?	22
COLUMNS?	6

Question	Your Response
Is your input a virtual matrix?	YES
TYPE DIMENSION SIZE OF YOUR MATRIX. (FORM: ROWS, COLUMNS)?	This will depend on the file. For ROSES it is 22, 6
LAST DATA ITEM READ FROM INPUT FILE?	
ENTER PRINT OPTION? (0 = NO PRINT, 1 = PRINT ENTIRE MATRIX, 2 = PRINT SUBMATRIX)	TO LIST FILE TYPE 1
DO YOU WISH TO CHANGE OR EDIT YOUR DATA?	NO

The computer will then print out the results for all variables and ask for directions.*

Question	Your Response
ENTER ANALYSIS TO BE PERFORMED?	FINISH

*An example of the results

COLUMN

```
    1            2            3            4            5          6
190.0000     72.0000      28.0000      20.0000      68.0000      1.0
140.0000     66.0000      27.0000       0.0000      87.0000      1.0
185.0000     69.0000      47.0000       7.0000      71.0000      1.0
128.0000     60.0000      48.0000       0.0000      70.0000      4.0
137.0000     62.0000      47.0000       0.0000      65.0000      6.0
165.0000     64.0000      51.0000       0.0000      65.0000      1.0
210.0000     74.0000      28.0000      30.0000      70.0000      1.0
124.0000     70.0000      21.0000       0.0000      65.0000      3.0
128.0000     68.0000      22.0000       0.0000      68.0000      2.0
 97.0000     60.0000      24.0000       0.0000      40.0000      1.0
152.0000     65.0000      39.0000       8.0000      72.0000      2.0
129.0000     52.0000      25.0000       0.0000      62.0000      1.0
204.0000     72.0000      31.0000       6.0000      80.0000      2.0
250.0000     70.0000      35.0000       0.0000      70.0000      2.0
123.0000     62.0000      32.0000      10.0000      70.0000      1.0
156.0000     65.0000      46.0000      20.0000      75.0000      1.0
192.0000     73.0000      28.0000       0.0000      67.0000      1.0
240.0000     73.0000      34.0000       0.0000      67.0000      1.0
128.0000     65.0000      24.0000       0.0000      67.0000      2.0
120.0000     64.0000      29.0000      10.0000      65.0000      1.0
127.0000     62.0000      46.0000       0.0000      75.0000      9.0
130.0000     66.0000      25.0000       0.0000      98.0000      5.0
```

PRINTOUT FOR ELEMENTARY STATISTICS

```
VARIABLE      MAXIMUM      MINIMUM       MEAN        STD. DEV.     STD. ERROR      RANGE

1            250.0000     97.0000     157.0455      41.7378       8.8985         153.0
2             74.0000     52.0000      66.0909       5.3711       1.1451          22.0
3             51.0000     21.0000      33.5000       9.7627       2.0814          30.0
4             30.0000      0.0000       5.0455       8.4204       1.7952          30.0
5             98.0000     40.0000      69.8636      10.5257       2.2441          58.0
6              9.0000      1.0000       2.2273       2.0686       0.4410           8.0
```

APPENDIX III

```
ENTER ANALYSIS TO BE PERFORMED.
? HISTOGRAM

DO YOU WISH TO READ NEW DATA ?  NO

ENTER VARIABLE NUMBER ? 3

TYPE A PRINT CHARACTER FOR THE HISTOGRAM ? *

ENTER THE NUMBER OF INTERVALS.  (MAXIMUM = 10) ? 5

                HISTOGRAM FOR VARIABLE  3

FREQUENCY    7    6    3    0    6
  --------+----+----+----+----+----+----+----+----+
    7        *
    6        *    *              *
    5        *    *              *
    4        *    *              *
    3        *    *    *         *
    2        *    *    *         *
    1        *    *    *         *
  --------+----+----+----+----+----+----+----+------
        21.000    33.000    45.000
             27.000    39.000    51.000

DO YOU WISH TO PRINT MORE HISTOGRAM ? YES

ENTER VARIABLE NUMBER ? 3

TYPE A PRINT CHARACTER FOR THE HISTOGRAM ? *

ENTER THE NUMBER OF INTERVALS.  (MAXIMUM = 10) ? 10

                HISTOGRAM FOR VARIABLE  3

FREQUENCY    4    3    4    2    2    1    0    0    5    1
  --------+----+----+----+----+----+----+----+----+----+----+----
    5                                                *
    4        *         *                             *
    3        *    *    *                             *
    2        *    *    *    *    *                   *
    1        *    *    *    *    *    *              *    *
  --------+----+----+----+----+----+----+----+----+----+----+----
        21.000    27.000    33.000    39.000    45.000    51.000
             24.000    30.000    36.000    42.000    48.000
```

INTERACTIVE COMPUTING IN TEACHING STATISTICAL CONCEPTS

Z. A. Karian and A. Sterrett
Denison University
Granville, Ohio 43023

I. INTRODUCTION

The level of mathematics required to explain concepts and prove theorems in an elementary statistics course is beyond the experience of most students. To avoid a 'cook-book' approach to teaching and learning statistics, a concept-oriented laboratory has been developed at Denison University. Individual data for approximately thirty laboratory exercises are generated either by students with a table of random digits or by a computer. This paper deals exclusively with those exercises which require the use of computers.

Each student is required to analyze different data sets during an assignment, and consequently must eventually do his own work. We encourage cooperation among students in the laboratory exercises. In many exercises the student responses are accumulated and used in class discussions, for example, in testing a certain statistical hypothesis. When only numerical responses are required of the student, say the equation of a regression line, the exercise is graded by the computer. This will explained more fully in section II.

We believe that our laboratory exercises help us meet the three goals that CUPM [1] recommends be incorporated into an elementary statistics course:

Clarification of key concepts in the course. In an elementary statistics course, many of the important ideas are difficult to explain because they are based on mathematical concepts. The notion of an unbiased estimator, for example, often is avoided in an elementary course because of its relatively sophisticated definition. In our course, a computer supplies students with indi-

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

vidual, finite populations from which students calculate parameters such as mean, median, or variance. The computer also supplies all possible samples of a fixed size from which students compute the average of various statistics to determine whether the average value equals that of a specified parameter. By this analysis students obtain insight into the concept of an unbiased estimator. Other ideas such as long-run relative frequency interpretation of probability, the meaning of the Central Limit Theorem, confidence intervals and significance level are also clarified through the laboratory exercises.

Performance of routine calculations. Applications of statistical procedures often require analysis of large amounts of data, more than can conveniently be handled with a hand-held calculator. Our students obtain individual data sets from a computer and use a statistical package to calculate means, variances, regression and correlation coefficients, t-values and the like. These exercises add realism to the course while simultaneously acquainting students with the utilization of statistical packages.

More active student participation in the development of statistical concepts. Each student receives a different data set from the computer and is asked to test a statistical hypothesis at a large significance level, say 20%. Each student comes to his own conclusion based on individual data, but in class we calculate the percentage of students who rejected the hypothesis which we now tell them is true. They observe that it is not unusual for five or six in a class of thirty to reject a true hypothesis as predicted by the theory and that we can influence the probability of making a Type I error by modifying the decision rule. The collective involvement of our students in testing a statistical hypothesis leads to a better understanding of basic principles by individuals.

Our students' understanding of the binomial model also is enhanced by their active involvement with an experiment. A computer provides each student with 50 samples of 7 random digits. Each digit represents the result of a phone call by a pollster to a randomly selected household: 0,1,2,3,4,5,6 or 7 designates that a television set in the household is not tuned in to Tuseday Night at the Movies; 8 or 9 designate that the set is tuned in to the movie. With his own data, each student has the experience of classifying each outcome of a binomial experiment as 'successs' or 'failiure' and counting the number of successes. Each student is asked to make a histogram of the 50 sample values and learns first-hand of the general shape of the binomial distribution. Of course, the experiment reinforces the idea that experimental results vary but that a suitable model can predict long-run results. The idea of a sampling distribution is illustrated by asking students to classify means of samples taken from rectangular distributions or values of $(n-1)s^2/\sigma^2$ where s^2 is the sample variance of a normal population having variance σ^2. With the aid of a computer, our students receive frequent exposure to the notion of sampling distribution as they work with their own data.

II. SOFTWARE

The software consists of essentially five programs each with a number of embedded subroutines. All programs are written in the BASIC-PLUS language and implemented on a PDP 11/45 computer. The programs perform the following general functions:

Data Generation. Similar but distinct data are generated for each student for each of the computer-based laboratory exercises. Two programs, DATBNK and DATA are involved in this process. DATBNK creates and maintains a 100X100 virtual core array (disk resident matrix). Each variable in the array consists of up to 100 entries and occupies a row of the array. The row location of the variable specifies the manner in which the data are manipulated before they are given to students. Since the nature of the data and their use is determined by the instructor, students can not access this program.

The program DATA is used by students to generate individual data for a variety of exercises. The similar but distinct data are generated by a set of algorithms which randomly perturb the fixed data created by the instructor or generate new data by a random process. In either case, the random number generator used in this process is seeded by the student data number so that a given student (with an assigned data number) will consistently get the same data; but other students will receive different data. The use of these programs is illustrated in section III.

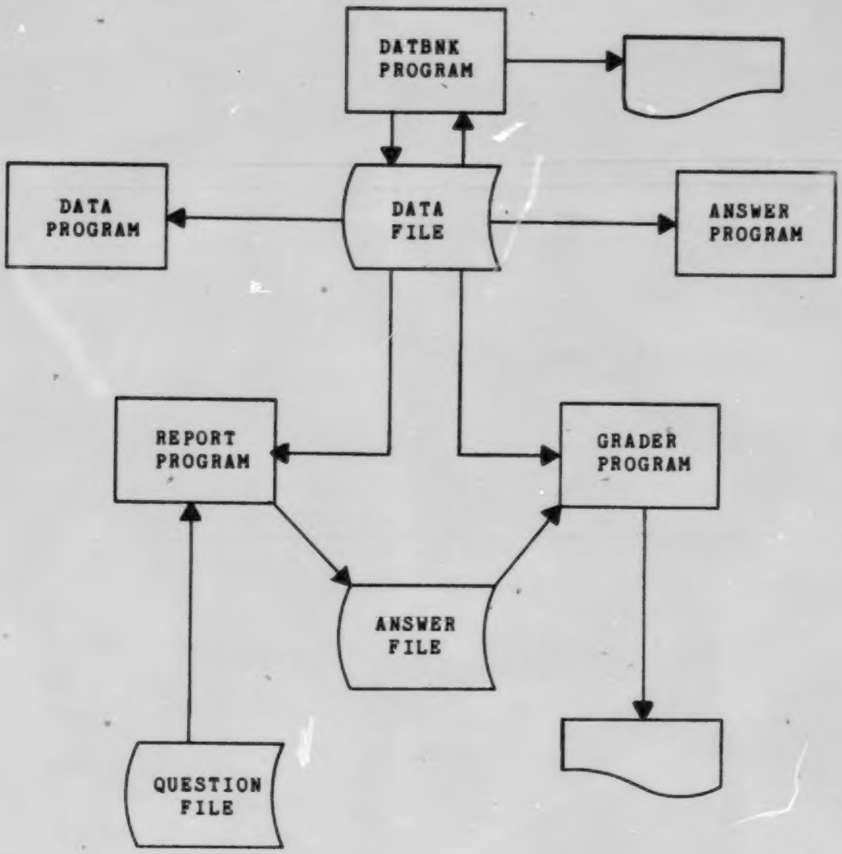
Answers. An obvious difficulty in giving students individual data (particularly in computational exercises) is the grading of the exercises; or at least informing the student of the correct answers after his work has been completed. The ANSWER program computes and prints answers for all computer based exercises for each student. To do this, the program uses the virtual core file created by the DATBNK program and regenerates the data supplied by the DATA program and proceeds to compute and print answers. This is done in a loop so that the process is repeated for each student. Needless to say this program is not available to the student.

Answer Reporting. To reduce the amount of paperwork created by the laboratory exercises, the REPORT program was written to accept answers to most exercises directly from the students. More importantly, the program is designed to be generous enough to inform students of errors (without divulging the correct answers) up to three times without penalty. On the third attempt the student is given the correct answers and further attempts by the student are not permitted. To allow more than three attempts without jeopardy will encourage guessing and sloppy work by students. Additionally, the instructor can set a deadline for a problem so that students must report their answers by that date; otherwise the program will not accept the answers.

To enable the REPORT program to ask the proper questions for a problem, an ASCII file, QUEST.FIL is created which contains all relevent information. This provides the flexibility of editing this file as changes and addition are made to the laboratory exercises.

The answers given to the REPORT program are stored in an answer file in a binary format which subsequently is used by the GRADER program to prepare reports and grades for each student.

The information flow between various components of the software is shown in the following figure.


DATBNK PROGRAM
DATA PROGRAM
DATA FILE
ANSWER PROGRAM
REPORT PROGRAM
GRADER PROGRAM
ANSWER FILE
QUESTION FILE

III. ILLUSTRATIONS

The examples below show the way the laboratory exercises and the programs are used in an attempt to implement the CUPM goals discussed earlier.

Example 1: Rigorous discussions of unbiased estimators is beyond the scope of an elementary statistics course. The concept, however, is an important one and students need to have at least an intuitive understanding of it. This exercise clarifies the concept and provides more active student participation in the development of the concept (the first and third CUPM goals).

EXERCISE: The purpose of this exercise is to investigate which of several estimators are unbiased estimators of the mean, median and variance of specific finite populations. Request variable SAMPLES from the DATA program to obtain a population of six nummbers and all (20) samples of size 3 selected without replacement.

a) What is the population mean, mu?
b) What is the population median?
c) What is the population variance?
d) Find the expected value of the sample mean
e) Find the expected value of the sample median
f) Find the expected value of the sample variance
g) Is the sample mean an unbiased estimator of mu?
h) Is the sample median an unbiased estimator of mu?
i) Is the first sample observation an unbiased estimator of mu?
j) Is the sample median an unbiased estimator of the population median?
k) Is the sample variance an unbiased estimator of the population variance?

Submit your answers to the REPORT program. NOTE: For parts g), h), i), j), and k) respond with a "1" if your answer is a "yes" and with a "0" if your answer is "no". If more than one of the estimators investigated in parts a), b) and c) are unbiased estimators of mu, which one is "best"? What is your criterion?

The student obtains the data for this problem as follows:

```
RUN (3.5) DATA
VARIABLE NAME?  SAMPLES

                        DATA 2

          POPULATION:  5  2  10  20  4  6

                       SAMPLES

[5 2 20]  [5 10 20]  [2 10 20] [5 2 4]  [5 10 4]  [2 10 4]
[2 20 4]  [10 20 4]  [5 2 6]   [5 10 6] [2 10 6]  [5 20 6]
[2 20 6]  [10 20 6]  [5 4 6]   [2 4 6]  [10 4 6]  [20 4 6]
[5 2 10]  [5 10 20]

DO YOU WANT MORE DATA?  NO
READY
```

The underlined parts here are what the student types at a terminal; the rest is printed at the terminal by the program.

The instructor can obtain the answers to this exercise by using the ANSWER program. Again the user responses are underlined.

```
RUN ANSWER
```

```
OUTPUT FILE?  KB:
LOWER AND UPPER DATA NUMBERS?  1,3
THIS PROGRAM WILL GENERATE THE FOLLOWING ANSWERS:

      1) MEANS AND ST.  DEV.'S REGRESSION  LINE  AND  CORR.
         COEFF.
      2) MEANS AND STANDARD DEVIATIONS FROM GROUPED DATA
      3) FREQUNCIES WITHIN GROUPS OF  RANDOM  INTEGERS  AND
         EXPECTED VALUES
      4) POPULATION MEAN AND ST.  DEV.  AND MEANS OF SAMPLE
         MEANS, VARIANCES, AND MEDIANS
      5) MEANS, ST.  DEV.S, CHI-SQUARE STATISTIC, T- OR  Z-
         VALUES, CONFIDENCE INTERVALS, AND A/R
      6) MINIMAL SAMPLE SIZE FOR SPECIFIED ALPHA AND BETA
      7) CONFIDENCE INTERVALS FOR THE BINOMIAL P USING  THE
         P  AND Q FROM THE SAMPLE AND THE PQ=1/4 APPROXIMA-
         TION
      8) ERRORS DUE TO ROUNDOFF AND THE CENTRAL LIMIT THEO-
         REM

WHICH OPTION DO YOU WANT?  4
VARIABLE?  SAMPLES


                   SAMPLING PROBLEM FOR VARIABLE SAMPLES

DATA ****POPULATION****      μ      σ²      V(x̄)    E(ξ)    V(ξ)    E(σ²)
 NO


 1   18 16  9  6  5  3     9.50   31.58   6.32    8.70   15.61   37.90

 2   20 10  6  5  4  2     7.83   35.47   7.09    6.10    4.29   42.57

 3   16 15 11  8  6  1     9.50   26.92   5.38    9.90    9.69   32.30
```

Students report their answers by using the REPORT program as follows:

```
RUN (3,5) REPORT

LAB. EXERCISE NUMBER? 14
SECTION NUMBER? 2

**** ATTEMPT 1 ****

MU? 7.83
CORRECT!

POPULATION MEDIAN? 5.5
CORRECT!

POPULATION VARIANCE? 39.72
YOUR ANSWER IS WRONG!

EXPECTED VALUE OF THE SAMPLE MEAN? 7.83
CORRECT!
```

```
EXPECTED VALUE OF THE SAMPLE MEDIAN? 6.10
CORRECT!

EXPECTED VALUE OF THE SAMPLE VARIANCE? 67.43
YOUR ANSWER IS WRONG!

IS THE SAMPLE MEAN AN UNBIASED ESTIMATOR OF MU? 1
CORRECT!

IS THE SAMPLE MEDIAN AN UNBIASED ESTIMATOR OF MU? 0
CORRECT!

IS THE FIRST SAMPLE OBS. AN UNBIASED ESTIMATOR OF MU? 1
CORRECT!

IS THE SAMPLE MEDIAN AN UNBIASED EST. OF THE POP. MEDIAN?1
YOUR ANSWER IS WRONG!

IS THE SAMPLE VARIANCE AN UNBIASED EST. OF THE POP. VAR.?1
YOUR ANSER IS WRONG!

READY
```

On the second attempt the result would be something like this

```
RUN (3.5) REPORT

LAB. EXERCISE NUMBER? 14
SECTION NUMBER? 2

**** ATTEMPT 2 ****

POPULATION VARIANCE? 35.47
CORRECT!

EXPECTED VALUE OF THE SAMPLE VARIANCE? 35.47
YOUR ANSWER IS WRONG!

IS THE SAMPLE MEDIAN AN UNBIASED EST. OF THE POP. MEDIAN?0
CORRECT!

IS THE SAMPLE VARIANCE AN UNBIASED EST. OF THE POP. VAR.?0
CORRECT!

READY
```

On the third attempt,

```
RUN (3.5) REPORT

LAB. EXERCISE NUMBER? 14
SECTION NUMBER? 2

**** ATTEMPT 3 ****

EXPECTED VALUE OF THE SAMPLE VARIANCE? 56.78
```

```
YOUR ANSWER IS WRONG!
CORRECT ANSWER IS 42.57

READY
```

After students have submitted their answers to the REPORT program the instructor can get a report or grade the students through the GRADER program.

```
RUN GRADER

WHICH SECTION? 2

THE FOLLOWUNG OPTIONS ARE AVAILABLE:

     1) INITIALIZE FILE
     2) SET DUE DATE FOR A PROBLEM
     3) GET REPORT ON A PROBLEM
     4) GET REPORT TO DATE
     5) EXIT

WHICH OPTION? 3

             REPORT ON PROBLEM 14 ON 16-Mar-78

                     DUE DATE: 3 / 10

DAT.NO.      ATTEMPT          RESPONSES         GRADE

  1             2            11011011111         8.18
  2             3            11111011111         9.09
  3             1            11111111111        10.00
  .             .                 .               .
  .             .                 .               .
  .             .                 .               .
```

The sequence of eleven 0's and 1's under the response heading is a binary code which indicates the correct (1 entries) and incorrect (0 entries) responses. According to the responses made by the student with data number 2 in the earlier illustration, all but the sixth part of the problem were answered correctly. This results in the response code 11111011111 for data number 2 and a grade of 10(10/11)=9.09. For uniformity all problems are graded on a 10 point basis.

Option 4 of the GRADER program, illustrated below, yields a more comprehensive report of the class performance on the laboratory exercises by listing in the same binary format the responses and grades for all relevant problems. The entry 14/3(11111011111)/9.09 listed under data number 2 refers to problem 14. The 3 indicates the number of attempts made, the sequence of 0's and 1's in parentheses is the binary code of responses and the 9.09 is the grade for that problem.

```
  DATA NO.= 1            TOTAL= 122.56            AVERAGE= 7.66

2/2(1111)/10          3/1(11111111)/10         4/3(01)/5
5/2(0101010)/4.29     7/1(1111110)/8.58        10/3(01010101)/5
12/1(111111)/10       13/2(11111111101)/9.09   14/2(01101101111)/7.27
19/3(111111)/10       20/2(110111)/8.33        21/3(110)/6.67
22/2(110111)/8.33     23/3(110111)/8.33        24/2(010111)/6.67
27/3(111100)/6.67


  DATA NO.= 2            TOTAL= 137.09            AVERAGE= 8.57

2/1(1111)/10          3/2(11111111)/10         4/2(10)/5
5/3(0111011)/7.14     7/1(1111111)/10          11/1(11110111)/8.75
12/3(011100)/5        13/3(01101011010)/5.45   14/3(11111011111)/9.09
19/1(111111)/10       20/1(111101)/8.33        21/1(111)/10
22/1(111111)/10       23/1(111111)/10          24/1(111111)/10
27/3(111011)/8.33
       .                     .                       .
       .                     .                       .
       .                     .                       .
```

__Example 2:__ In contrast to the previous illustration, this exercise is conceptually a 'typical' problem in elementary statistics. In this case, however, we are able to provide students with realistic data without concern for computational complexity.

> EXERCISE: request variables WEIGHT and BLD PR using program DATA. The variable WEIGHT measures the weights of individuals who were subjects in a 1952 study of heart attacks in Los Angeles. the variable BLD PR measures the diostolic blood pressure of these same individuals. Find the Y-on-X regression line using variable WEIGHT for X and BLD PR for Y. Specifically compute:
>
> a) Mean of Weight
> b) Standard deviation of Weight
> c) Mean of BLD PR
> d) Standard deviation of BLD PR
> e) Slope of the regression line
> f) The Y-intercept
> g) The correlation coefficient

To get the data the student reuses the DATA program.

```
RUN (3,5) DATA
VARIABLE NAME?  WEIGHT
                              DATA 2
                              WEIGHT
          194     226     182     155     183     176     153
          168     137     123     156     170     152     149
          142     163     178     224     235     136     164
          170     159     195     153

DO YOU WANT MORE DATA?  YES
```

```
VARIABLE NAME?  BLD PR
                                DATA 2
                                BLD PR
                84    94    84    78    117    89    96
                77    83    83    84    95     82    83
                71    95    83    91    116    70    74
                99    82    77    79

DO YOU WANT MORE DATA?   NO

READY
```

As the data indicate, hand calculations are likely to be too cumbersome. We do not suggest any computational method in the problem, but we have shown our students the use of a standard statistical package. They may choose not to use this computational aid at their own risk.

The use of the ANSWER, REPORT and GRADER programs is similar to what has already been illustrated in the previous example.

IV. CONCLUSIONS

Two out of 30 laboratory exercises were illustrated above. The REPORT and GRADER programs accept and grade 15 of these problems. During a one semester course 20-25 of these exercises are assigned. Of these, 15 are 'graded' by the GRADER program. We offer two or three sections of elementary statistics each semester with approximately 35 students in each section.

A timesharing computing environment, although not essential, is highly desirable for the type of application we have described. Students need frequent and virtually unlimited access to a computer. Our computing facilities consist of a PDP 11/45 computer with 104 K words of memory, 120 million characters of on-line storage and 20 terminals dedicated to instructional use and available 24 hours a day for a student body of 2150. We will supply documentation and the BASIC-PLUS programs to interested individuals.

REFERENCES

1. Committee on the Undergraduate Program in Mathematics, Introductory Statistics Without Calculus, Mathematical Association of America, 1972.

2. A. Sterrett and Z. A. Karian, A Laboratory for an Elementary Statistics Course, American Mathematical Monthly, VOL. 85 2(1978).

THE EFFECT OF CAUSE ON INSTRUCTIONAL COMPUTING AT A LIBERAL ARTS COLLEGE

David D. Dexter

Alma College, Alma, Michigan 48801

Introduction

As part of its long-range plan, Alma College established in 1975 the following institutional goals for program improvement in science education:

1. Upgrade the competencies of our undergraduates in the natural and social sciences and mathematics.
2. Maintain or improve the professional capabilities of our faculty.
3. Increase the application of computer technology in the teaching of all science disciplines.

It was recognized that undergraduate education is incomplete without exposure to and experience with modern computing machinery and that computer utilization should be significant in the curriculum.

Although substantial resources were to be expended in the process, having arrived at this point, the next steps followed directly. A decision to acquire new computing equipment, through private means, was reached. Support for using the new hardware in the instructional arena was sought from the National Science Foundation under the CAUSE (Comprehensive Assistance to Undergraduate Science Education) program.

Then, in 1976 the CAUSE grant was awarded and a PDP-11/70 was installed.

Since a major portion of the CAUSE grant was earmarked for support of instructional computing applications, this communication describes our strategies for utilizing this support and relates some initial classroom experiences. I will begin with a few words about Alma College, comment on our instructional computing history, move on to the CAUSE supported program and conclude with some thoughts for those who would learn from our experiences.

Alma College: The Institution

Alma College is a four year, undergraduate, liberal arts college located in the center of Michigan's lower peninsula. Established in 1886, it emphasizes personal attention and broad-based quality education. The student body of 1100

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

is a highly qualified (average high school g.p.a. of 3.48, 85th percentile of ACT scores) group, representing more than one-half the states and several foreign countries. Many are in preparation for professional or graduate schools.

Computing History

In the early 1960's limited use was made of computer hardware available in the community. These applications tended to be in the nature of special projects and were limited to a small number of students (perhaps five per term). By the late '60's, educational computing needs had outstripped the local supply of surplus cycles and telephone lines to a commercial time-sharing facility were being used. Still, only a few students (perhaps 20 per term) could be involved because of high cost and inconvenience. In 1971 a 40K IBM 1620 was installed on campus. Programming courses (in FORTRAN) were introduced and several simulations were implemented. These activities reached about 50 students per term but further expansion was hampered by the limitations of the machine.

In the Fall of 1976 our current computing hardware was installed. Details of the configuration can be found in Table 1. The PDP-11/70 operating under RSTS/E served some 500 students in its first fully operational term.

Computational Aspects of the CAUSE Program

Concurrently with the acquisition of the PDP-11/70, Alma College received a CAUSE (Comprehensive Assistance to Undergraduate Science Education) Grant from the National Science Foundation. This grant supports a program designed to improve science education and has at its core educational computer applications. Specifically, funds were granted to implement a program of supervised training for the faculty of the natural and social sciences in the use of the computer and in the development of educational projects using the computer.

A computer applications coordinator was hired to direct two summer workshops and to assist with the implementation of instructional projects. The program which was offered to our science faculty was as follows:

1. Participating faculty would be paid to attend a one-month workshop which would examine instructional computer applications in general and our computing system in particular.
2. Each participant would agree to use the computer in some way in at least one course. He would plan and prepare such materials as necessary to properly integrate his project into the course. If desired, a participant would be supplied with programming help.
3. After a year a second workshop would be offered for assessment and further development.

Faculty were asked to "apply" for this program by preparing a plan suggesting computer use in a course for which they were responsible. Fifteen science faculty responded affirmatively. Their distribution by department is shown in Table 2.

The 1977 Workshop

The calendar for the 1977 workshop is summarized in Table 3. We organized the workshop in three units. Since both the hardware and its native language, BASIC-PLUS, were new to the campus, we began with an intensive series of lectures and practice sessions. For instructors, we used three of the group who had already taught introductory programming courses and then, by scheduling four hours of meetings per day--and sustaining ourselves with liberal quantities of coffee and doughnuts--we progressed through assignment statements, branching, looping and simple file handling in five working days.

The next nine days were occupied with a sequence of eight invited speakers. We tried to invite congenial, experienced teachers from a variety of disciplines (which corresponded to the mix of our faculty's interests) with different philosophical approaches to instructional computing. The speaker list is presented in Table 4. Each speaker began with a discussion of his instructional situation: what problems he was addressing, his audience, his approach. This continued for as long as was appropriate and we then moved on to hands-on experience with the "software of the day" and in-depth discussions with individual faculty. Each speaker spent at least one day with us and we ate lunch as a group.

Before each speaker presented his material, his software was implemented in our system. With few exception, (there was only one 2:00 a.m. session) the imported programs were induced to run with little difficulty--believe it or not!

The last week of the workshop was devoted to discussions of what we had learned and the directions individual participants would take for their own classroom applications. Two of the speakers (from neighboring institutions) participated in these discussions to offer encouragement and guidance.

The directions chosen by the participants for their instructional computing efforts were as diverse as the individuals themselves. A summary of these projects is presented in Table 5. Some of these applications were coded during the workshop period. Some were written during the ensuing summer. Others remain unimplemented.

Fall Term, 1977 Experience

Who used the facility?

Instructional use of the PDP-11/70 in the 1977 Fall Term is summarized in Table 6. The precision of these numbers could conceivably be improved but we are comfortable with the accuracy of the 500 total class-students estimate. Seven departments and some 15 courses are represented.

What software was used?

Most of the students used materials obtained through the CAUSE program. The rest were concerned with programming courses or were preparing their own code to solve homework assignments. Table 7 indicates major software utilization for this one term. Four different courses in two departments made use of some parts of the DEMOS programs (from G.E.-TEMPO) for demography. The other popular package was the XTAB statistical routines developed by Zavin Karian at Denison University. Several classes used them to analyze data bases or survey data as class projects.

Evaluation. Or, Is the Computer Making the Instructional Process Better for Teachers and Students?

One term or one classroom application or one workshop does not a comprehensive experience make. While the total impact of this effort will not be known for some time, it will come as no surprise that we think we have some indications of what has worked and where problems lie.

The 1977 Summer Workshop

- The participants felt that ten lectures in five days was a bit too fast for a programming introduction. On the other hand, we were trying to provide an experience which was stimulating and unifying and which would help make the concepts and techniques the outside speakers would discuss more readily usable.

Most participants did not expand upon this introductory material. When we stopped talking, they stopped programming. It probably would have been wise to include one or two programming tasks to be completed during the remaining three weeks of the workshop.

The speakers were well received but most faculty would have preferred more "show" and less "tell". Most of the ideas for classroom applications, as outlined by faculty before the workshop, were not changed by the comments and suggestions of the speakers. It may not be particularly efficient to "reinvent the wheel" but we all at times need room to make mistakes in order to learn.

The last week of the workshop (which called for discussion of intended applications and some individual effort toward implementation) was not particularly successful. As a group we lacked the experience necessary to recognize the ramifications of a particular idea so constructive discussion of intended applications was limited. Further, the sudden shift from a structured to a non-structured schedule seemed to be unnerving, and little was accomplished outside of scheduled workshop activities.

Like a football team leaving town to prepare for the big game, it probably would have been better if the workshop had been held off-campus. The demands of home, family or office competed for our attention and distracted us from the task at hand.

We were surprised at the reluctance participants showed toward exploring instructional applications whose topic lay outside the limits of a particular discipline. We probably should have encouraged more hands-on experience by requiring that each day specific exercises be completed and a written evaluation be prepared.

The Fall, 1977 Term

"Body count" statistics are scattered through the preceeding text and it seems safe to say that the number of students who received some computer exposure increased significantly over previous levels. If every class had a computer component (a situation neither desirable nor practical) then the maximum audience for an instructional application program would be about 4 classes X 1100 students or 4400 class-students per term. If, for example, one-half of the classes "should" have a computer component then the 500 class-students logged in Fall, 1977 represent less than 25% of the potential audience. By any measure, we have a ways to go.

Has the experience been worthwhile? We surveyed 192 students in six courses from four departments with the instrument shown in Figure 1. To achieve some quantitative assessment of the results, I assigned a value of +1 to a response of "agree", -1 to "disagree", and 0 to "neutral" or "not applicable". Similarly, for question six, a vote for increased computer use received a weight of 1, present level, a weight of 0 and reduced level, a weight of -1. The most positive possible response would be scored +6 and the most negative response would be -6. Averages for the six classes responding to the survey are shown in Table 8.

Only one student in the group surveyed disagreed with everything! Eleven students handed down net negative assessments. The class average scores coincide with my qualitative evaluations of what happened in the terminal room. The lowest scores occurred in classes where either a major bug was discovered while trying to run the exercise or where preparation of the students was inadequate. The most favorable response came from a physical chemistry class where the students were already well motivated and computer utilization is a natural adjunct to normal course content. Even allowing for reluctance to down-rate course material, it appears that the students have not been turned-off by our efforts and are willing to sample further attempts.

So far, the science faculty seem to be pleased with what has transpired. Several students were moved to remark on the apparent enthusiasm of their instructor for the new tool. The real test will come when the CAUSE support runs out.

Plans For Further Evaluation

Continued evaluation of this project will take place in two general ways:

1. Our projects and ideas are continually examined by faculty and students alike. Most of our classes are small so that difficulties become apparent quickly and alternatives can be tried. Similarly, techniques which are successful tend to be used again.
2. For more formal evaluation, we have adopted the following approach: First, we have an in-house evaluator. This individual is a member of the Education department, outside the science divisions, and is experienced in evaluating educational effectiveness. Second, we have retained a consultant who is not part of the college. He has a great breadth of evaluational experience and a national perpective of instructional computing.

Both evaluators will be using similar tools: interviews with and opinion surveys of faculty and students, examination of course syllabi and catalog descriptions before and after the CAUSE program, review of student course evaluations and review of computer utilization.

We at Alma College are very much in the formative stages of instructional computing. The development process seems to be an evolutionary one but we feel our beginnings are encouraging.

Table 1

Computing System at Alma College
Fall, 1977

Hardware

PDP-11/70	96K	Memory
	1	RP04 Disc Drive
	1	TU16 Tape Drive
	1	CR11 Card Reader
	1	CM11 Card Reader
	1	LP05 Printer (300 lines/min.)
	9	Decwriter II Terminals
	2	VT52 CRT Terminals
	1	H9 Heath Video Terminal
	1	4012 Tektronix Graphic Terminal
	1	4662 Tektronix Flat-bed Plotter

System Software

RSTS/E
BASIC-PLUS
BASIC-PLUS-2
FORTRAN
RMS-11K

Table 2

Department Affiliation of 1977 Computer Applications Workshop Participants

Psychology:	1 (3.5)*	Political Science:	1 (2)
Chemistry:	3 (5)	Biology:	2 (3)
Sociology:	3 (4)	Mathematics:	3 (5)
Economics:	1 (2)	Physics:	1 (1)

*Number of faculty in that department.

Table 3

Summary Calendar for the Alma College 1977 Computer Applications Workshop

May 20, 1977 - May 30, 1977	Introduction to BASIC-PLUS and the PDP-11/70
May 31, 1977 - June 10, 1977	Invited speaker "Show-and-Tell"
June 13, 1977 - June 17, 1977	Digestion, Discussion, Decision
June 20, 1977 - June 22, 1977	Attendance at CCUC/8

Table 4

Invited Speakers
for the
Alma College

1977 Academic Computer Applications
Workshop

Tuesday, May 31, 1977

Zavin Karian - Department of Mathematics, Denison University, Granville, Ohio.
Mathematical Sciences; Statistics and Quiz Generation.

Wednesday, June 1, 1977

John Palmisano - General Electric, TEMPO, Washington, D.C.
DEMOS (Demographic/Economic Models of Society); A Computerized Learning System.

Thursday, June 2, 1977

William Bewley - Control Data Corporation, Arden Hills, Minnesota.
Experiments in Cognitive Psychology.

Friday, June 3, 1977

David Howard - Department of Economics, Delta College, University Center, Michigan. Pedagogical Computing in a Community College.

Monday, June 6 and Tuesday, June 7, 1977

David Cook - Department of Physics, Lawrence University, Appleton, Wisconsin.
Uses of Computer Graphics and Introduction to Graphics Software.

Wednesday, June 8, 1977

Ronald Collins - Department of Chemistry, Eastern Michigan University, Ypsilanti, Michigan.
Tutorials and Test Generation; Institutional Sources: CONDUIT

Thursday, June 9, 1977

Ronald Rosenberg - College of Engineering, Michigan State University, East Lansing, Michigan.
Simulation in Environmental Systems.

Friday, June 10, 1977

Thomas Evans - Department of Chemistry, Denison University, Granville, Ohio.
Simulation of Chemical Reactions and Tutorials.

Table 5

Summary of Faculty Projects for Instructional Applications

Design New Simulations
- Population Genetics (Sociology)
- Visual Stimulation (Psychology)

Computerized Test Generation
- Chemistry

Computer Managed Instruction
- Freshman Laboratory Records (Chemistry)
- Test and Quiz Score Analyses (Mathematics)
- Field Data Analysis (Biology)

Computer Assisted Instruction
- Topics in Behavior Modification (Psychology)
- How to Write Answers to Essay Questions (Political Science)

Computer Graphics
- Particle in a Box (Chemistry)
- Complex Analysis (Mathematics)
- Lecture Slide Preparation (Physics)
- Function Plotting (Mathematics)

Use Existing Simulations/Packages
- METROPOLIS (Sociology, Political Science)
- DEMOS (Sociology, Biology)
- SIMSTATE (Political Science)
- Survey Analysis (Sociology, Political Science)

Table 6

Instructional Applications: Number of Students Involved
1977 Fall Term

Department	Class-Students
Mathematics (including programming courses)	110
Psychology	25
Chemistry	15
Political Science	15
Sociology	175
Biology	160
Education	20
TOTAL	520

Figure 1

ALMA COLLEGE SURVEY OF COMPUTER USE IN SCIENCE COURSES
DECEMBER 1977

______________ COURSE

As you know, we have introduced or extended the use of our computer in several courses this term. We are attempting to evaluate the effectiveness of this computer usage and we need your help.

Please respond to each of the items in Part A, indicating with a (✓) your opinion regarding computer usage in this course. Also, please provide elaborating comments and any suggestions for improvement, on the reverse side.

Part A: Use of computers in this course:	AGREE	NEUTRAL	DISAGREE	NOT APPLICABLE
1. made the course more interesting				
2. assisted in problem solving, analysis, or development of other scientific skills				
3. assisted in learning concepts, principles, theories, or other course content				
4. added to my understanding of the use of computers in scientific inquiry				
5. added to my computer skills				

6. should be continued at: ( ) the present level; ( ) an increased level

-COMMENTS AND SUGGESTIONS ON REVERSE SIDE-

Table 7

Instructional Applications
Software Used
1977 Fall Term

(In order of estimated frequency of use)

Program/Package	Class-Students
DEMOS (Demography programs from GE TEMPO)	200
ECOTONE* (Field Data Analysis - Biology)	120
EVOLVE* (Population Genetics - Sociology)	50
Survey Analysis Routines	50
Assorted statistical programs	50
Computer Graphics (Chemistry)	8
TOTAL	398

*Developed by CAUSE workshop participants

Table 8

Results of Surveying
Computer Use in Science Courses
Fall Term, 1977

+6 implies strong acceptance of computer use; -6 implies strong rejection.

	Class Size	Average Response
1.	6	2.0
2.	9	4.8
3.	11	3.4
4.	41	2.6
5.	31	4.0
6.	94	3.5

DEVELOPMENT OF A COMPUTER ORIENTED PROGRAM IN SCIENCE AND MATHEMATICS (1)

Sheldon P. Gordon
Mathematics Department
Suffolk Community College
Selden, N.Y. 11784

1. Introduction

The CCUC meetings and Proceedings are undoubtedly the principal vehicles for the interchange of information on what people are doing in educational computing. The meetings which the author has attended were extremely valuable experiences in terms of determining the state of the art in a variety of different disciplines.

However, in discussions with other participants at the Conferences, it became clear that many had come hoping to find answers to such questions as: how do you organize an educational computer program?, how do you fund it?, how do you keep it viable? how do you expand an existing small program? what type of computing facilities would be appropriate?, and so forth. The present paper is intended to address some of these concerns by sharing the (probably typical) problems and experiences of the faculty at the author's school and by outlining the type of educational program which was developed as a solution. Since many of the problems we faced are common to most schools and especially to two year institutions and smaller colleges, it is felt that the solutions we found may be of interest to our colleagues by suggesting some useful ideas. This program, incidentally, is currently being implemented thanks to the very welcome support provided by the National Science Foundation under its CAUSE (Comprehensive Assistance to Undergraduate Science Education) program.

2. Background on the College

Suffolk County Community College is one of the largest community colleges in the country and is the second largest college in the 68-unit State University of New York system after the university center at Buffalo. The College is currently expanding into a multi-campus operation with two new campuses having opened within the last few years. The current enrollment, across the three campuses, is almost 20,000.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

The College very definitely exists to serve the needs of its students and the community. 98% of the students live in Suffolk County. Upon graduation, over 95% of those who obtain employment do so in Nassau and Suffolk Counties, which comprise Long Island. This area is one of the most technologically developed sections of the Country as measured by the size and number of engineering and related firms located here. In a recent survey we conducted of personnel directors of these firms, the overwhelming reply was that these companies are seeking potential employees who have some experience with computers.

At present, the College's sole computer resource is an IBM 370/138 system, using only batch processing. This system is devoted primarily to administrative work and a data processing program, though the County Legislature is also tied in to it. Needless to say, this arrangement has not been conducive to providing either the opportunity or the facilities to develop a computer oriented program in our science courses.

Furthermore, the central campus of the College is located within five miles of SUNY at Stony Brook and a large percentage of our graduates transfer to the University to complete their studies. Many others transfer to one of the eight other colleges (mostly private) within 25 miles of the College. For the most part these students came to us from local high schools which have computer programs based on time sharing facilities. Further, when they transfer from Suffolk to Stony Brook or one of the other four year institution, they again encounter extensive interactive computer usage. The lack of such facilities at SCCC has created a serious two-year hiatus in the educational development of these students.

As the College's President, Dr. Albert Ammerman, expressed it in the CAUSE proposal, "the College feels that it has become incumbent upon itself to provide the maximum exposure to and experience with computers to its students. In fact, were we not to do so, we would be doing our students a major disservice and would be ignoring our raison-d'etre".

3. Project Objectives

In order to address the needs of our students, especially those in the sciences and related fields, we intend to develop and implement a computer oriented program in a wide variety of courses covering six major academic areas: Mathematics, Physics, Chemistry, Earth and Space Sciences, Engineering and Economics. We anticipate that when this program is in full operation, in about two years, it will affect as many as 10,000 students per year, though of course there will be considerable duplication among people taking concurrent or consecutive computer oriented courses.

The program itself is in two parts. First, in order to implement it at all required the acquisition of an interactive computer system to complement the College's IBM 370/138. To satisfy this essential need, part of the CAUSE grant was intended for the purchase of a PDP 11/34 minicomputer system providing BASIC, FORTRAN and CAI capacity and supporting 14 CRT terminals, some with graphics capability, and printers. The CPU is being housed in the present Computer Center and the entire system will be maintained by the staff there. Most of the terminals are located in various buildings on the central Selden Campus and several are located at the two new campuses and connected to the CPU via leased data telephone lines. The system was chosen to allow for the eventual expansion to about 32 terminals, since we expect heavy usage in the areas mentioned plus a movement toward computers in several other academic areas.

The terminals are available both in special rooms and in some 35 classrooms which have been specially wired to provide in-class use of the computer.

At one time, if this system had been made available to us, we would have been overjoyed. Fortunately, we realized in time that, in the long run, merely obtaining the hardware would have been disastrous. At present, only about a dozen faculty members at the College have the knowledge and experience needed to fully utilize such a system. Had we gotten just the hardware, then, it would have essentially sat around collecting dust.

Historically, many institutions have fallen into this trap. They have acquired, often at great expense, sophisticated computer systems and have counted on nothing more than the curiosity of their faculty to lead to large scale implementation of the computer into the curriculum. To this day, colleges both large and small are still repeating this essentially fruitless and sterile approach.

Accordingly, in order to have a successful program, it is also essential to involve a large number of instructors and this can only be done through a formal training program. In addition, if they are then to actually use the computer in their courses, it is essential to have adequate course materials available which are geared specifically to their courses and to their students. We anticipate that about 65 of the 85 instructors in the six areas will become involved in the program.

However, any program of this type costs money and would not be possible without some outside funding such as the CAUSE grant.

Thus, the second (and in many ways the more important) facet of our project is the educational program. Under the CAUSE grant, we will be able to provide released time (3 credits each) to some 25 faculty members. This will allow them the opportunity to reevaluate our present courses with a view toward incorporating the computer and to prepare and develop packages of materials for each course. The type of package needed will consist of a series of 10 to 15 modules. Each module will consist of appropriate computer programs for each specific topic with accompanying descriptive material and problems for student use. In addition, a manual for instructor use will also be written as part of the package.

As of this writing, we have people preparing packages in calculus, finite mathematics, survey of mathematics, general chemistry, general physics, and computer usage in chemistry labs. Moreover, several other individuals have proposed packages for descriptive statistics and for other aspects of general physics.

Furthermore, we will be offering a series of summer workshops to acquaint large numbers of instructors with programming and with the specific applications of the computer in their courses via the packages previously developed. In addition, we have already offered several mini-workshops for faculty who have had a little experience with computers, but who require a refresher in programming and an introduction to the new computer system.

This type of faculty training is essential for the success for any type of large scale program in computer usage. For the most part, especially at small colleges and at two year colleges, the faculty have little or no experience using computers and generally stand in awe of them. Many will be extremely fearful of displaying ignorance and so will be unwilling to utilize computers in class until they feel on firm ground. Mere intellectual curiosity is not sufficient impetus for most people to adopt the computer in their classes. Rather, a system of incentives and a training program are required to overcome most of the barriers. In addition, it should be expected that there will always be a group of faculty at almost any school (hopefully a small group) which will never become reconciled to the computer.

Our plan is to encourage extensive use of the computer in as many subject areas and courses as possible by providing the opportunity, the training, the assistance and the encouragement needed. For incentives, we are offering released time for preparing packages of materials and modest honoraria for attending the various workshops. However, we do realize that there will be many different philosophies regarding the ways in which the computer can be used in science education. As a result, we intend to allow the faculty working under us to have the freedom of choice to incorporate the computer in whatever ways they feel is most appropriate.

Thus, in the more advanced courses, the computer will be introduced as an exceptionally useful tool and educational device for both simulation and problem solving. It will enhance the students' understanding of the material, will allow them to be exposed to more realistic problems and situations than the artificial classroom cases usually covered and will better prepare them for subsequent transfer to four year institutions.

In those courses not specifically intended for science majors, the computer can be used in a variety of different ways. In some courses, it will serve as a simulator for modeling different situations. In other courses, the computer will be incorporated as a computational device or to perform repetitive operations. In a general survey of mathematics course, the students will learn how to program the computer, rather than simply calling for library programs. Whichever mode is used in a particular course at this level, the emphasis will be on developing computer literacy so that the students will gain an appreciation of the potentials of the computer and, perhaps more importantly, its limitations.

We expect this exposure to and experience with computers will encourage many more students to enroll in the more advanced programming courses offered by our Data Processing Department. In addition, by providing such a large number of students with BASIC as a common and easily learned programming language, we are presenting the entire college with the opportunity to build on our program and make use of the conceptual and computational powers of the computer in a far wider range of subject areas than is covered in the present project. Finally, this program should also significantly enhance the students' career opportunities either through transfer to four year institutions or through better employment prospects upon graduation.

Some of the specific details of how the computer will be used in the various academic areas are as follows:

In mathematics, the computer will be incorporated into a full spectrum of subjects ranging from liberal arts survey courses up through the calculus sequence and on to courses in differential equations and linear algebra. The type of usuage of the computer will reflect the modes most appropriate to the individual course. Thus, in the statistics courses or in finite mathematics, it will serve primarily for simulation game theory and, to a lesser degree, as a computational aid for handling data, performing matrix operations, and so forth. In precalculus, calculus and differential equations courses, it will be used as a conceptual device to portray the behavior of functions, solutions of equations, limits, derivatives, definite integrals, approximation techniques, etc., as well as a tool for computation and problem solving. In survey of math courses, the computer will be introduced as an independent topic.

In Physics, Chemistry, Earth and Space Sciences and Engineering, the computer will serve in a variety of capacities. As a simulator, it will enable the students to examine the underlying scientific principles of a given situation without concern for the often formidable mathematical analysis associated with it. In this way, students can be exposed to subjects they would otherwise not be mathematically equipped to handle for several years. Similarly, as a problem solving device, the computer allows students to cope with a variety of problems normally too complicated for classroom consideration. Further, in laboratory courses, the computer can serve as a pre-lab simulator to insure that the student has a clear grasp of the experiment before physically performing it. Also, it will provide on-the-spot checks on the experimental data collected or alternatively, it can be used to generate the data needed as part of the experiment. Finally, many students come into the various introductory science courses with deficiencies of various types and the computer can provide remediation via a CAI mode.

In Economics, the computer will be used primarily in simulating different models such as macroeconomic systems, market equilibrium models, monopoly models and the like. It will make the students active participants in economic decision processes. Further, the immediate feedback will provide an excellent learning device to demonstrate the problems faced by a policy maker in managing a modern economy.

In order to avoid the time involved in teaching BASIC programming in each of the different courses, not to mention the duplication of effort, we will develop and offer seven-week mini-courses in BASIC which will be given twice a semester. This approach has several major advantages over a full-term course. If taken during the first half of a semester, a student can use it as a corequisite for any computer oriented course and take it at a time when he is not overly concerned with term papers, exams and the like. If taken during the second half of a semester, it will serve as a prerequisite for courses the following semester and will be an attractive option to students who have withdrawn from another of their courses and who require extra credits to maintain academic standing.

The overall responsibility for the project lies in the hands of the author as Project Director. In addition, since the scope of the project is so wide and it involves so many people from different academic areas, it is necessary to have a number of assistant directors, each responsible for two of the subject areas. It is their job to solicit proposals for possible packages from interested faculty, to assist and oversee the actual implementation of the packages and to assist in evaluating the success of the program. They are also responsible for developing sample packages in their own areas to serve as models.

The summer workshops for faculty training will be held for three weeks each summer. The first week will be devoted to teaching the BASIC language to the participants and to giving them an overview of the computer, its capabilities, and the use of the various types of terminals. During the following two weeks, the various assistant directors will work directly with the participants from their own academic areas. The second week will be devoted to demonstrations of precisely how the computer can be incorporated into courses in each subject area and the participants will become familiar with the materials already developed. Finally, during the third week, each participant will be expected to develop individual modules in his or her own area. In addition, should there be a demand for it, more advanced programming techniques in either BASIC or FORTRAN will be introduced on an individualized basis.

We expect that most of the participants will be sufficiently excited by the workshops and intrigued by the potentialities of the computer itself as an educational tool to devote time during the balance of the summer to gaining extra experience on the computer.

Further, we expect these people to try out the materials developed during the following academic year. In addition, we anticipate that, as a result of the workshops and classroom experience, we would then have a considerably enlarged pool of available talent on which to draw for the preparation of additional course packages. In turn, these people and the material they develop could be used during the subsequent summer's workshop to expand the audience still further. It is this chain reaction process which will generate the interest, participation and support on the part of the faculty which will guarantee a successful program.

We have also considered the fact that many computer packages are available through such groups as CONDUIT, Project COMPUTE, various publishers, user groups and from the hardware manufacturers themselves. However, our goal is to get the faculty as deeply involved in this program as possible. We feel that this can be best achieved by having faculty members develop their own packages of programs as much as possible. For one, the resulting materials will reflect the philosophies prevalent on our own campus. Second, there will always be people on campus who know all of the details of the programs written and will be available to demonstrate and assist their colleagues in their use. Moreover, there is a certain satisfaction and, above all, a feeling of confidence gained from doing the work oneself which we feel greatly outweighs the possible loss in elegance or sophistication of the finished product. Having the most efficient computer programs available does not necessarily mean that people will use them. On the other hand, if instructors have developed their own, they will naturally want as many other people as possible to use them. This type of infectious enthusiasm and personal involvement are other major factors we are counting on to result in a successful implementation of the program.

Admittedly, the above philosophy may be regarded by many as being tantamount to 'reinventing the wheel'. However, we feel that it is essential if a large number of faculty are to become involved in actually using the computer. In many ways, learning to work with computers is comparable to learning a foreign language. However, no one ever seems concerned about the need to begin with a primer in French, say, before a person is ready to read Les Miserables. In particular, we feel that this approach is especially necessary in a community college or small college where the faculty do not have the background in research which might make a move towards computers somewhat easier from the point of view of personal confidence.

On the other hand, once we do have a large pool of experienced personnel available, it will make sense to then acquire outside packages with the expectation that such materials would be used.

As indicated above, our primary objective in this program is to involve as many faculty members as possible in using computers in their courses. The greater the faculty participation, the greater the subsequent impact on our student body. However, when we consider the wide range of different courses in the six areas which are likely to be affected by the program it became painfully obvious that evaluation of the effectiveness of the project can not be measured by student response or performance. Rather, we decided that success of the project will be dependent on the degree of faculty involvement. To that end we devised a detailed questionnaire for all 85 faculty members who are covered under the project. This instrument essentially is intended to determine the "computer literacy" of each respondent and allows us to categorize each instructor at a variety of levels of both computer experience and degree of use of computers in courses. The information we have collected represents the base line data for our evaluation. By the end of the three years, we will conduct a comparable survey and determine on the basis of it how much the general level of computer experience and course-related usage of computers by the instructors has increased.

In summary, by the end of the three year grant period, we expect as many as 10,000 students per year to be affected by the program. They will have been exposed to the computer, will have gained varying degrees of facility in its use, and will have been the recipients of the enhancement in understanding and problem solving ability available through it.

The effects on the departments concerned and on the school in general will also be significant. The fact that a program of this magnitude was developed and successfully implemented at the College will provide an inestimable boost to the confidence and morale of the faculty. This will encourage people in many areas, not just the ones covered in the project, to develop new programs and to feel that they too will be successful and worthy of outside support. The long range total effect on the College as a whole will be immensely positive and fruitful.

Footnote

(1) The author gratefully acknowledges the support of the National Science Foundation under grant #77-05817

COMPUTER RESOURCE CENTER--A CONSOLIDATION OF INSTRUCTIONAL DEPARTMENT TERMINALS

Victor Langer

Milwaukee Area Technical College

INTRODUCTION

In a classroom situation an instructor with 15 to 30 students per section is limited to the experiences the students can be given at one single computer terminal. The problems in increasing the number of terminals became a budget and policy question at the Milwaukee Area Technical College (MATC), which has utilized computers in instruction since the 1950's and in CAI since 1971. MATC is principally a post-secondary school offering 141 occupational programs to over 15,000 FTE students on four campuses. The programs include associate degree (both general education transfer and terminal occupational), apprentice, adult vocational (both diploma and certificate occupational programs), ABE, and adult high school. In addition, the college owns and operates two public television stations, Channels 10 and 36, which serve the community in cultural and educational programming including College of the Air.

MATC runs a Honeywell 6060 computer and has a Datanet 355 to handle all of the time-sharing users. The Datanet 355 has twenty low-speed telephone-connected lines, twelve 1200 baud hard-wired lines, twelve 300 baud hard-wired lines, and four high-speed asynchronous Remote Job Entry lines. The 6060 has 256 K words of memory (1,536,000 characters), and the on-line storage capability comprises six 191 drives, giving in excess of 750,000,000 characters of on-line storage. In addition, there are five-multi-density nine-track tape drives. The system operates under the control of GCOS.

CRC--A BEGINNING

Instructional departments had long realized the necessity of having computer terminals available for students, especially in the areas of mathematics, science, industrial technologies, and business. All too often a department obtained a terminal which either became overworked or actually became logistically unavailable for all students in a class. Therefore, with the requests for second and more terminals the computer center staff began to be concerned about the impact on the central processor and on budgets. Most instructional departments were turned down on requests for second and more terminals, and single-terminal departments actually under-utilized the terminal due to the lack of accessibility by students. In order to have some sort of guideline for placing terminals, a policy was established at MATC requiring a minimum usage of

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

40 hours per month of student terminal time. Also, a central area was set up by the Educational Media Department to house terminals from departments which would give up a single terminal in order to gain multiple-terminal access for their students. In the 1974-75 year, six departments volunteered their terminals to start what has become the Computer Resource Center. In addition, several electronic calculators, including programmable calculators, were brought there for checkout or use in the center. This center became available on a scheduled basis for students of any teacher. Also, unscheduled terminals were available on a first-come, first-served basis. Many teachers began bringing classes for a mini-course in computer literacy, other teachers began setting up drill exercises in the basic skills areas and in the vocabulary areas of the many technical subjects offered at MATC, some teachers used the center as backup terminals for data processing courses and computer science courses, some teachers had their classes solving technical problems by using the terminals, and some developed simulations of business and technical operations. An individualized course, "Computer Applications to Instruction," was developed to provide the faculty with a basic understanding of computers, programming in the language BASIC, and opportunities to explore applications for their courses. This course helped many teachers to prepare their own software, but for most teachers the course established an ongoing dialog with the systems analyst and other service personnel.

CRC INSTRUCTIONAL SOFTWARE DEVELOPMENT

Any faculty member has available the assistance of complete software development, including a library of 1,200 programs obtained from other schools. About one half of the 400 on-line programs currently in use were developed by MATC. The center has developed some general retrieval programs which allow faculty members to submit instructional statements that can be put up on the computer in just a few hours. Once a teacher begins using computer support and finds it to be a valuable contribution to learning, the teacher usually continues to improve the application in cooperation with the supportive staff, including a media specialist, systems analyst, and programmers. The staff works with the instructor to upgrade the initial drill-type program to keep the level of program sophistication in line with the instructor's needs. The use of additional media, simulations, case study, and even managed instruction are all available on a developmental basis. The Achievement Testing Department provides a statistical analysis of test scores, assists with the preparation of tests from test question banks, and performs a textbook computer reading-level evaluation. Various instructors have developed record-keeping software for their individual use.

CRC STUDENT USE

The CRC is open from 7:30 a.m. to 5:00 p.m. Monday through Friday and in 1976-77 served over 4,600 student contacts in 24 departments, with over 2,500 hours of terminal time for just six terminals. This is a significant increase from the previous year, 1975-76, with approximately 3,400 student contacts in 17 departments, with approximately 1,900 hours of terminal time.

The students entering the center who are in need of a terminal for a technical course usually have been instructed on terminal use and are self-sufficient in operating the terminal. Students coming in for drill exercises or simulations usually need to be signed on the terminal, must have the programs called, occasionally are tutored, and need to be signed off. The mechanics of the terminal should not interfere with learning.

It is important that the CRC educational assistant is able to make the student feel at ease and provide assistance as necessary. The CRC is the communication center for the teacher where the educational assistant takes the prescriptions from the teacher and reports back the activities and progress of the student. A phone call, a visit, or a note from a teacher or student can reserve a terminal or all terminals to support the classroom needs for a computer. a class can take over the entire center, if arranged.

CRC FACULTY SURVEY

The overall reaction of the faculty to the CRC in a recent survey of 32 user instructors reflected a need for more terminals and availability during evening hours. Most teachers surveyed indicated their use of the CRC has been principally a supplemental drill-type learning activity. Some of the teachers depend on the CRC to provide computer activities for their students in order to complete a course. Examples include computer literacy (BASIC programming), problem solving, drafting or art type computer graphics, simulation (some in a mini-course format), etc. As a result, the CRC is becoming more cost justifiable than just an add-on improvement of instruction-type service. The CRC has not researched the significance of the improvements in instruction, but the many faculty comments would indicate a high "plus" value. For example, a basic education teacher reported that the computer drill exercises removed the tedious repetitive activity from the classroom and allowed more classroom time to deal with applications of the skills. A reading teacher claims that she can teach more things to more students. A math teacher uses the CRC as a lab and claims the computer is a valuable motivator. Some technical instructors claim computer simulations increase the number of decision experiences before performing the real situations, which has improved the model performance in a shorter time. Physical therapy instructors found that students can deal with anatomy terms more effectively in a lab in their first semester physical therapy courses after using the computer drills. Also, some technical instructors found that vocabulary drills help students to prepare for civil service exams and other types of certification testing.

CRC STAFFING AND FUNDING

The one CRC educational assistant, with help from student aides when available, is responsible for tutoring students, maintaining the terminal schedule, gathering data about program and terminal usage, assisting with instructional programming, issuing reports, and taking care of minor maintenance and inventory, besides being the humanistic link between students and hardware. The instructional systems analyst also assists during peak periods and covers for lunch hours, etc., which allows for observation of students while they are learning. This feedback helps in revision and creation of new programs.

The CRC is located at the Downtown Milwaukee Campus, and the fall of 1977-78 the six terminals were increased to eight, and terminals were added to each library to serve students at the three other campuses. The library staff from each campus was trained in computer programming (language BASIC), signing on and off students, scheduling the terminals, working with teachers, reporting back use to the central CRC, and referring program development activities back to the CRC.

At the present time all costs of the CRC are carried as overhead, and individual departments are not charged for service. When a department has its own password, the department is charged for computer time only as a benchmark figure to compare all users. The computer is owned by MATC and may at some future time be cost accounted to the user.

CRC--A FUTURE

A recent award by the NSF will expand the CRC into developing computer graphics for courses in architectural technology, mechanical technology, computer science, civil technology, electrical technology, visual communications, and physics. The grant will provide four additional terminals, which will increase the center to twelve terminals, plus one graphics terminal and a plotter at the downtown center. Also, a micro-processor is to be used as an interface between CRT terminals, a printer, and other audiovisual devices. The CRC is also acquiring a micro-processor to find ways to relieve the large-scale central Honeywell 6060 of drill exercises and problem solving.

The opening of the Computer Resource Center has given much impetus to the faculty at MATC in developing software for instructional purposes. Formal recognition by Local 212 of the AFT has made the more than 600 full-time teachers aware of the benefits of computer-augmented instruction. A slide-tape program. "The Case of Computer No-Put," was developed to take away the mysteriousness of the CRC and the CAI in workshops for faculty groups. The 8-minute program was done to the theme of Dragnet, and a follow-up demonstration of applications apparently succeeded in motivating new users, as the growth rate was 35% last year.

The individual departments that gave up a single terminal in order to have access to several terminals at one time have received terminal use and software development support far exceeding expectations. The CRC has become a success in making more use of terminals and providing a center for leadership in computer applications to instruction.

A COMPUTER SIMULATION FOR TEACHING CLIENT REFERRAL SKILLS

William Abernathy, Ph.D.*

Elizabeth McBride, B.A.

Ohio University-Chillicothe

ABSTRACT: *A computer simulation is described that is designed to assist in the teaching of referral skills to community mental health workers and others in the human services. Three components of successful referral are the focus of the simulation. The student must "interview" the "client" and obtain information necessary to the referral. He must identify those agencies that service the simulated problem and he must select the agency(s) for which the client is eligible to receive services. The program randomly generates client descriptors and problems based on statistical profiles of the local catchment area. These profiles may be modified for other users.*

An important function of the community mental health professional is that of a community services "broker". A recent social work text defines the role of broker as "...a guide in the 'civilized jungle' of the service network. His function is to put people in touch with community resources that they need but do not know about" (Brieland, Costin, and Atherton, 1975).

"Hotlines", referral centers, storefront information, and other information clearinghouses serve essentially as brokers for clients in need of community service information. However, all mental health professionals must assume the broker role from time to time and should therefore acquire at least minimal referral skills.

**Dr. Abernathy is an Assistant Professor of Psychology, Ohio University-Chillicothe, P.O. Box 629, Chillicothe, Ohio 45601. Ms. McBride is a Data Analyst and Programmer at Ohio University-Chillicothe.*

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

An important component of the referral process that is often overlooked in the formal training of community mental health workers is a working knowledge of the specific services and eligibility requirements of the various community agencies. Casework instruction has proven quite valuable but too time consuming to present a variety of problems and agencies. What is needed is an adjunct to casework methods that provides the student an alternative to rote memorization of referral manuals or on-the-job failures that may do a disservice to his clients.

OBJECTIVES OF CLIENT REFERRAL SIMULATION

The computer simulation to be described was designed to provide students in a two-year human services technician program an interactive learning environment for developing a working knowledge of specific services and eligibility requirements of community agencies in south-central Ohio. The simulation was not designed to replace existing instruction but only to assist the student in three components of the referral process.

First, the student must "*interview*" the "*client*" and obtain information necessary to selecting the appropriate agency(s). Secondly, based on the interview the student must *identify* those agencies that provide services that meet the client's needs. Finally, the student must *select* from among the identified agencies the agency for which the client is eligible to receive services.

DESCRIPTION OF THE SIMULATION

The client referral simulation is written in extended BASIC. The program has run for the past two years on a Data General Nova 2 minicomputer and ASR33 teletypes as interactive terminals. The program may be considered in terms of four sections including the area agencies, client description, referral interview, and the referral. Figure 1 presents a simplified flow chart of the program.

Area Agencies - The services and eligibility requirements of the sixty most utilized agencies in the south central Ohio area were obtained from the *Ross County Directory of Health and Welfare Services* (1975) which was expanded or revised in some cases through information provided directly by local agencies. Based on the directory each agency was listed in the program under one or more service categories (see Table 1). A second listing in the program consisted of categories by eligibility (see Table 1).

By modifying print statements in the program the user would be able to tailor the agency names, services, and eligibility to his particular community. A second advantage of the program over texts or manuals is the capacity for continual revision, as various agency parameters change, without developing an entirely new program.

Client Description - This section of the program was added after several trial runs. It consists of a brief description of the client including randomly selected characteristics (see Table 1). Actually, the description plays no part in the referral process but was added to provide the simulation a more "human" quality.

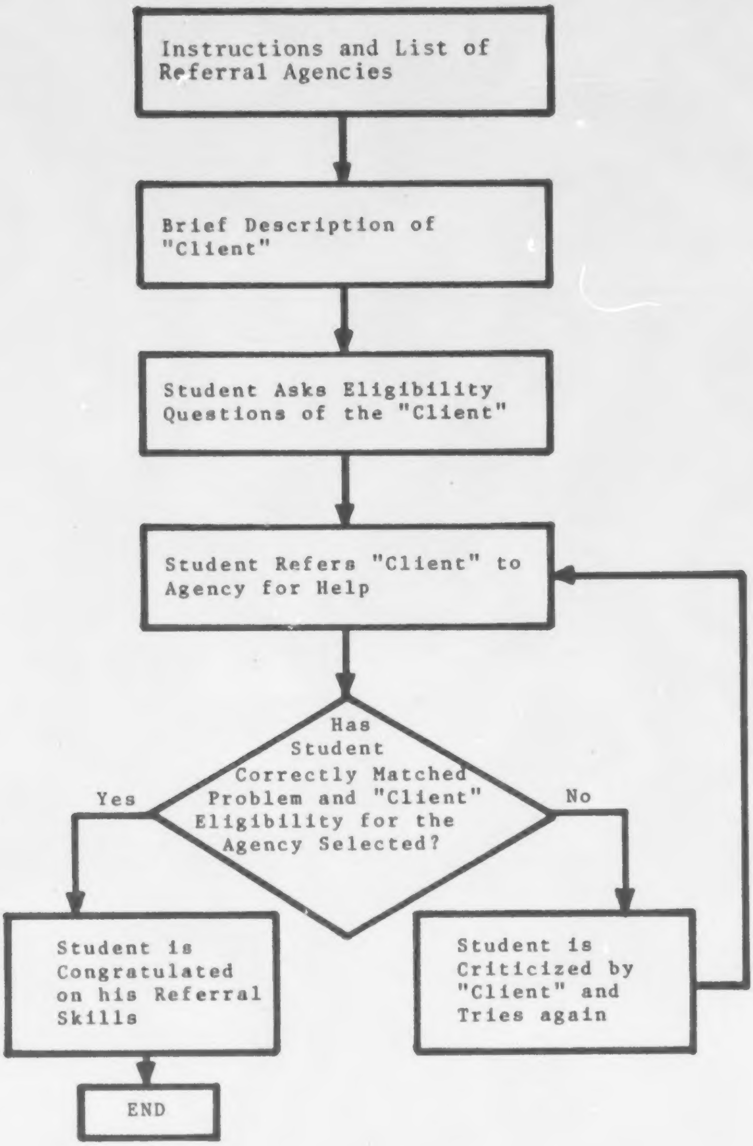

Instructions and List of Referral Agencies
Brief Description of "Client"
Student Asks Eligibility Questions of the "Client"
Student Refers "Client" to Agency for Help
Has Student Correctly Matched Problem and "Client" Eligibility for the Agency Selected?
Yes
No
Student is Congratulated on his Referral Skills
Student is Criticized by "Client" and Tries again
END

Fig. 1 Flowchart of "Client" Referral Simulation

Table 1 Client Referral Simulation Section Listing

Agency Service Categories	Eligibility Categories	Client Description
alcohol problems	age	sex
extreme emotional problems	marital status	aptitudes
mild emotional problems	residence	interests
drug problems	employment	adjustment
handicaps	income	appearance
legal problems	education	social relations
family problems	military service	
vocational counseling	dependents	
vocational training	health	
financial problems	transportation	

Referral Interview - Following the agency listing, instructions, and the client description the computer prints a ? and waits for a one word question that will assist the student in referring the client. A number of alternatives for each question are acceptable as inputs. For example, to obtain information as to whether the client is employed, any of the following inputs are accepted: "employment", "employed", "job", "working", or "work". If a word is entered that is not in the program, or is misspelled, the computer responds with "I do not recognize that word, try another". Each acceptable input results in the computer searching the eligibility files and selecting an appropriate print statement.

Eligibility files contain a number of print statements each with an associate probability. For example, if the student enters "income" the computer searches the "income" file. The possible print statements and their probabilities are: "I earn almost $2000 (.02), $3000 (.03), $4000 (.04), $5000 (.05), $6000 (.06), $7000 (.07), $8500 (.14), $12000 (.24), $15000 (.28), $20000 (.06), or "over" $25000 (.04). Therefore, over a number of runs the computer will, for example, generate a client that earns $12000 a year approximately 24% of the time. The print statements and probabilities for each eligibility file were developed using the 1970 Census and Population Characteristics of the Population, Ohio and the Statistical Analysis of the Paint Valley Catchment Area (1976).

To determine the nature of the problem (which is effectively another eligibility requirement) the student inputs "problem". The computer then randomly samples the "agency service" category file and selects one of the ten problem areas to print. It may, on a random basis, repeat this process and select from one to three problems on any given run. Other users may be able to refine this section by changing these probabilities to match the actual percentages of their population.

A final important feature of the interview is that failures to ask questions about any aspect of eligibility result in the computer "assuming" the client is ineligible. This feature may require the student, after several unsuccessful referral attempts, to further interview the client to establish eligibility in earlier neglected categories.

Referral - When the student feels he has sufficient information to make a referral he types "ready" to which the computer responds "With what problem are you dealing?" The student types in the problem area listed in the interview. For multiple problems only one problem is handled at a time. Once the problem is typed the computer responds with "To what agency will you refer your client?" The student then types in the code of the agency he feels serves the problem and for which the client is eligible.

Four possible outcomes for each referral are possible. If the selected agency provides the service and the client is eligible the computer "thanks" the student and either asks for a solution of its other problems, or where all problems are solved, asks if the student wants another client or is finished.

If the selected agency does not service the problem the computer prints "Thanks for nothing. They don't handle my problem." Following this statement the student is asked to try another referral.

If the selected agency handles the problem but the client is not eligible to receive their services the computer prints "I wasted a lot of time going there. I'm not eligible to receive their services. Would you suggest another agency?"

A low probability statement that is randomly selected is "The agency you referred me to can't help me at this time. Would you suggest another?" If this statement occurs the student may input the agency a second time and succeed in making the referral.

In some cases client eligibility may be produced that precludes a successful referral or the student may be unaware of services provided by some agencies. If the student is unable to successfully refer the client he may type "defeat", escape from the program, and initiate a new client.

SAMPLE RUN

The following is an actual computer run of the simulation. The number 69 refers to the "Pickaway-Ross County Vocational School". The client was ineligible because he had completed high school. The number 67 refers to the "Chillicothe branch of Ohio University".

The following is a description of your client:

Your client is a female. Client appears to be average in intelligence. Client's main interest is agricultural. Psychological adjustment appears to be average. Client is averagely dressed. Socially the client seems to be shy.

During this interview session please ask one word questions that will help you in making your referral; e.g., Age (remember to push orange return key).

When you feel that you have all the necessary information and are ready to make your referral type Ready.

? Age

I am 31 years old.

? Kids

I have 3 children.

? Live

I live in Washington Court House in Fayette County.

? Problem

I have a vocational training problem.

? Employed

I am employed.

? Income

I earn about $8500/year.

? Car

I have transportation.

? Service

I am not in the service nor have I been.

? School

I have completed high school.

? Married

I am married.

? Ready

If at any time you feel that you need to ask your client more questions, type More.

```
     With what problem are you dealing?

     Watch your spelling.

     ? Vocational Training

     To what agency will you refer your client?

     ? 69

x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x

     I wasted a lot of time going there.  I'm not eligible to
     receive their services.  Would you suggest another agency?
     If at any time you feel that you need to ask your client
     more questions, type More.

x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x-x

     With what problem are you dealing?

     Watch your spelling.

     ? Vocational Training

     To what agency will you refer your client?

     ? 67

*****************************************************************

     I went where you suggested and they are going to help me.
     Thanks for your understanding and congratulations on your
     exceptional referral skill.

*****************************************************************

     Would you like another client?  Type Yes or No.

     ? No
```

COMMENTS

The simulation has been tested on 78 students to date. Modifications to the program were made as problems and questions arose. The final group of 18 students did not report any significant difficulties in either following the instructions or making successful referrals.

Each student was required to complete three runs. However, many students elected to complete a number of runs beyond the requirement indicating the simulation was interesting if not enjoyable.

Due to the continuous modification process as deficiencies were discovered, no comparative data is presently available as to the value of the program in teaching the skills objectives for which it was designed. The authors hope others will request copies of the program and perform evaluations in other locales, programs, and with other student populations.

REFERENCES

Brieland, D., Costin, L.B., and Atherton, C.R., Contemporary Social Work: An Introduction to Social Work and Social Welfare. New York: McGraw-Hill, 1975.

Nineteen Seventy Census of Population Characteristics of the Population, Ohio. Washington: Bureau of the Census, U.S. Government Printing Office, 1973.

Ross County Directory of Health and Welfare Services. Columbus, Ohio: The Mid-Ohio Health Planning Federation, 1975.

Statistical Analysis of the Paint Valley Catchment Area. Chillicothe, Ohio: Paint Valley Mental Health and Retardation Board, 1976.

THE APPLICATION OF SIMULATION TECHNIQUES TO ENHANCE STUDENT UNDERSTANDING OF COMMERCIAL BANK MANAGEMENT

Alan K. Reichert and N. J. Greenberg

Indiana University-Purdue University, Fort Wayne, Indiana

Introduction

Important changes taking place in the financial sector during the past decade and a half have significant implications for the conduct of macro-monetary policy, the efficient allocation of resources among sectors, the structure, stability and overall performance of the financial industry. Major technological developments are in the initial stages of remaking and modernizing the flow of funds process and the exchange of financial information between firms. Recent capital market and regulatory concern regarding excessive growth, untested structural change, and imprudent management strategies in a period of increased economic stress and uncertainty underscore the need for more enlightened commercial bank management. Of primary importance is the need to develop a general appreciation of changing economic trends and their subsequent impact upon the complex set of interrelationships which define the bank management process.

The purpose of this paper is to describe the current use of a computer assisted bank management simulation game designed to reinforce and enhance more traditional approaches employed in teaching basic bank management principles. The main advantages associated with such a simulation include the opportunity to actively participate in a management game which explicitly recognizes the effects of competition and uncertainty in the decision making process. The opportunity for immediate feedback serves to reinforce the learning process and stimulates greater student interest.

The paper is organized according to the following format. Section I presents a brief overview of the Oklahoma Bank Simulator (OBS), highlighting important student decision requirements and il-

---

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

lustrates the basic operating structure of the model. Section II provides a description of the Commercial Bank Management Course (F-446) as presently being taught here at Indiana University-Purdue University (Fort Wayne). A detailed discussion of the instructional objectives and operational use of the simulation game is provided along with a student/instructor evaluation of the model's performance, plus suggested modifications for future use. Section III presents the results of a recent survey of current OBS users in an effort to identify and evaluate the models performance in a variety of instructional applications employing a unique combination of instructional objectives and operating procedures. A further discussion of recommended, as well as in-process modifications, is provided. Section IV provides a number of conclusions regarding the relative effectiveness of the model in alternative instructional environments and offers various recommendations for enhancing future performance.

SECTION I

MODEL OVERVIEW

The Oklahoma Bank Simulator (OBS) is a deterministic multiperiod management simulation game designed to model the key management decisions required of a commercial bank operating in a highly competitive, predominantly rural environment. OBS illustrates many of the complexities and interrelationships involved in bank management, since it requires a careful analysis of the bank's liquidity position, sources and cost of funds, plus a critical assessment of alternative investment opportunities - all within the context of a dynamic, competitive environment subject to continually changing economic conditions.

At present the OBS illustrates competition between three rural commercial banks operating within a single county assumed to represent the relevant market area. Version 3 of the model permits the addition of up to eleven counties (each containing three competing banks), but assumes no inter-county competition. The simulation is presently designed for a maximum of six periods of play, with each period representing a six-month short-run planning horizon.

A significant number of exogenous economic variables are included to realistically illustrate the many environmental factors outside the control of commercial bank management. This exogenous set includes factors such as county personal income and retail sales, indices of farm prices and range conditions, various money market measures such as the Federal Funds and "prime" loan rate, certain regulatory constraints relating to required reserves and capital adequacy, plus historical loan/deposit market information. In addition, a likely range of values for the following decision period is forecast for such variables as county personal income and retail sales, farm prices, prime rate, and interest rates for a number of alternative investment securities.

The management team for each bank is required to make decisions regarding (1) pricing policy relating to interest rates on four categories of loans, time deposit and service charge rates, loan officer salaries, and advertising expenditures, and (2) investment policy relating to the optimal holdings of loans and investment securities. Lending opportunities include ten-year agricultural real estate loans with ten equal principal payments, one-year agricultural production loans with a single lump-sum principal payment, a two-year commercial loan with two equal annual principal payments, and a two-year consumer installment loan with four equal semi-annual principal payments. Investment opportunities include six-month, one, and three-year taxable federal securities, plus two and five-year non-taxable municipal bonds. (See Table I for a listing of relevant variables.)

For computational purposes, two distinct levels of analysis are distinguished in the model: 1) aggregate market (county) financial data, and 2) individual bank market share relationships. Market volume supply and demand equations are used to estimate the aggregate supply of demand and time deposits for the entire county. Variables used to estimate these supply relationships include county personal income, the levels of lagged demand and time deposits, seasonal and trend factors, yields on time deposits, and alternative exogenous investment opportunities.

The individual aggregate loan demand equations for agricultural real estate and production loans, commercial, and consumer installment loans are determined by key economic variables such as county personal income and retail sales, average annual interest rates charged by all three competing banks on each class of loans, index of farm prices and range conditions, seasonal and trend variables, plus a lagged stock variable for each of the loan categories.

Having calculated the total market supply of demand and time deposits, and the total market demand for each category of loan, these aggregate market volumes are then distributed among the three competing banks according to exogenously estimated deposit and loan market share equations. For example, the allocation of the endogenously determined fixed supply of demand and time deposits is influenced by both the absolute and relative changes in service charge, time deposit rate, advertising expenditures plus the lagged percentage of market share for each type of deposit. The market share equation for each category of loan is a function of changes in bank decisions in both absolute terms and in relation to comparable competitive bank decisions in regard to advertising expenditures, interest rate charges on loans, and the individual bank's lagged loan market share.

Provisions in the model allow banks to sell at discount previously acquired loans and investments beginning with the oldest earning assets in the bank's portfolio.

An important element of commercial bank management involves adequate liquidity or proper cash management. A required reserve equation is included to reflect current regulatory requirements with both deposit size and structure instrumental in establishing the level of required reserves. Furthermore, a provision for minimum cash holding to meet basic operating needs is included. To the extent that a bank's current level of cash and deposits are insufficient to accommodate its loan and investment requests, operating cash needs, and reserve requirements, the bank is forced to meet such a deficiency by purchasing Federal Funds at the current interest rate.

Bank performance is reflected in a semi-annual income statement which includes interest income from loans and securities, plus service charge income on demand deposits. Major expense items include interest paid on savings deposits, loan officer salary expense, and "other" expenses which include estimated transaction costs for implementing various loan/investment decisions, depreciation and over-

TABLE I

Input and Output Data

Input Supplied by Participants	Exogenous Data Supplied by Game Administrator	Output Received by Participants
Time Deposit Rate	County Personal Income	Statement of Conditions (Assets, Liabilities, Capital Account)
Loan Officer Salaries	County Retail Sales	Loan Deposit and Capital Ratios
Service Charge on Demand Deposits	Index of Farm Prices	Income Statement (Income and Expenses by Category and Type of Asset)
Advertising and Promotion Outlay	Index of Weather Conditions	Summary of Decisions
Interest Rates on Loan	Interest Rates on Investment	Portfolio Maturity Table
Maximum Volume of Loans	Interest Rate on Federal Funds	New Loan Volume
Purchases of Government Securities	Prime Interest Rate	Market Economic Conditions
Purchases of Municipal Bonds	Market Volume of Loans in Previous Periods[1]	Market Average Interest Rates on Loans and Deposits
	Market Volume of Deposits in Previous Periods[1]	Market Share of Loans and Deposits
	Market Share of Loans in Previous Period[1]	Forecasted Economic Conditions[2]
	Market Share of Deposits in Previous Periods[1]	Expected Interest Rates on Investments and Federal Funds[2]
	Forecasted County Personal Income[2]	
	Forecasted County Retail Sales[2]	
	Forecasted Index of Farm Prices[2]	
	Forecasted Prime Interest Rate[2]	
	Expected Interest Rates on Investments and Federal Funds	

---

[1] After the first decision these data are calculated within the program and stored for use in future periods.

[2] Ranges rather than single values are given for these variables to provide some indication of expected future economic conditions in the county

Source: Journal of Bank Research, Spring 1976, p. 59.

head expenses, plus any discount expense associated with the sale of earning assets. Income taxes are determined in accordance with existing corporate tax rates, with an appropriate adjustment for tax-exempt municipal bond earnings.

In addition, a capital requirements equation is included which explicitly considers differences in bank asset structure in the determination of an adequate level of bank capital. A realistic balance between long term sources and uses of funds is insured by introducing a maximum long-term loan to time deposit ratio.

Given the deterministic nature of the simulation design, OBS includes no explicit formal measure of risk or uncertainty. At the same time, the uncertainty associated with competitive bank policy and the need to make intermediate and long run investment decisions in the context of a changing economic environment, adds a considerable degree of "operational uncertainty" to the gaming process. As currently constituted, the model provides the following semi-annual output for each participating bank:

1 - Statement of Condition (balance sheet) with selected key operating ratios.
2 - Semi-Annual Income Statement indicating the level of forced Federal Funds borrowing (if any) and the accrued interest expense associated with such borrowing.
3 - A summary of the bank's decision input.
4 - A table of loan and investment maturities.
5 - The volume of new loans made by various loan categories.
6 - Data describing economic conditions within the county for the previous period.
7 - Average pricing decisions for deposits and loans for all banks within the county for the previous decision period.
8 - Loan and deposit market share performance for individual banks.
9 - Predictions for the next period applicable to investment security yields, plus an expected range of values for key economic variables.

SECTION II

COURSE APPLICATION

OBS was first introduced into the Management of Commercial Banks course (F-446) during the fall semester, 1976. The objectives of the course were to provide the student with the following:

1 - A basic understanding of the various types of decisions required of top level bank management.
2 - An appreciation of the many interrelationships between major decision variables.
3 - A working knowledge of the regulatory and competitive environment within which modern banks must operate.

F-446 is a senior level finance course where the typical student has successfully completed at least a year of basic accounting and economic principles, a number of finance and marketing courses, a basic computer systems course and one or more management simulation-based courses. During the course of the sixteen-week semester, the following basic topics were fully developed in lecture:

1 - The economic and institutional dimensions of both money and

capital markets.
2 - The evolution, objective, and evaluation of current bank regulations.
3 - An analysis of bank structure and performance.
4 - An analysis of economies of scale in banking.
5 - General theories of commercial bank management.
6 - Key determinants of bank profitability.
7 - Bank portfolio and capital management techniques.

The strong emphasis upon the economic and regulatory environment reflects the fact that bank management decisions are strongly conditioned by a host of external institutional and market forces.

The simulation model was introduced during the last five weeks of the semester and conducted concurrently with class lectures. Twelve students were enrolled during the fall semester, with four students assigned to jointly manage a single bank. Students were free to select their own team members and a team captain was selected to assume primary responsibility to see that basic administrative functions were properly being performed in a timely manner. The primary reason for encouraging team, rather than individual participation was to provide an environment which would promote considerable discussion and interaction among participants, and to more realistically model actual bank management conditions. Two practice play periods were conducted prior to the actual simulation run, to familiarize students with the basic mechanics of the game and to allow them to develop some "feel" for the responsiveness of key decision variables. The final competitive simulation involved six decision periods covering a period of approximately three weeks. Students were typically given three or four days to evaluate last period performance and formulate a set of new decisions.

Minimal class time was devoted to a review of individual bank performance, except to clarify operating or procedural questions. Instructor consultation with individual bank management teams outside of class was freely available on an "as needed" basis. Each team was provided with a reference manual containing key bank operating ratios, performance measures, and estimated operating costs for rural banks of similar size. Actual student consultation requirements were minimal during the semester (perhaps a total of two or three hours) since each student had substantial previous simulation experience and all were well acquainted with basic bank management concepts at that point during the course.

To encourage maximum student participation in the simulation game and provide students with an opportunity to formally analyze their bank's performance using the techniques and terminology presented in lecture, each student was required to submit a "project report" at the close of the semester. The report, which represented 25 percent of the final grade, included a statement of the bank's initial profit and market share objectives, the development of a set of specific operating strategies designed to accomplish those objectives, and a period-by-period analysis of actual bank performance and subsequent changes in strategies as required.

Student Evaluation

At the completion of the simulation project, students were asked to evaluate the degree to which they felt the simulation goals were achieved, the overall contribution of the model to the course objectives, and suggestions as to where the simulation could be improved. Five major instructional objectives were established for the simulation project. All students were asked to rank each objective according to the degree to which he or she felt the objective was accomplished, on an integer scale from one to five (with a rank of five indicating total compliance with the stated objective, and a one indicating little or no progress towards the stated goal).

Student Evaluation Results

SIMULATION OBJECTIVE	MEAN RANKING
A. The simulation is intended to provide a realistic learning experience in making important policy decisions regarding the acquisition and allocation of bank funds	3.4
B. The simulation should provide the student with an increased appreciation of the interrelationships and complexities associated with portfolio management	2.9
C. The simulation should illustrate the need for sound financial planning and analytical decision making	2.7
D. The simulation should generate productive discussion and interaction between participants during the decision making process	3.9
E. The simulation should serve to integrate the various theoretical dimensions and current principles of bank management presented in class in a realistic and practical manner	2.8

It is apparent that the simulation game was most successful in promoting student discussion and as a means of illustrating general bank principles. The results obtained when attempting to illustrate more complex, technical issues, appear to have been substantially less successful.

When asked to comment on a number of issues which the instructor had identified as possible opportunities for improvement, students responded in the following manner:

1 - Two thirds of the students felt the average loan rate and the general prime rate should be more closely related.
2 - Over ninety percent of the students felt that a more realistic sloped yield curve for the investment portfolio is warranted.
3 - Approximately sixty percent of the students felt that the model should be more sensitive to competitive bank pricing policies, seventeen percent felt the model should be less sensitive to competitive interest rates, while the remaining twenty-five percent were satisfied with the present degree of interest rate sensitivity.
4 - Over ninety percent of the students felt the model should be more responsive to both advertising and salary expenses, with top priority placed on correlating loan officer salaries with overall bank performance.
5 - Over ninety percent of the students thought it would be productive to explicitly introduce the concept of risk, in terms of a stochastic loan loss relationship.
6 - When asked to indicate the optimal number of periods of play for the final simulation run, responses ranged from a low of six to a high of thirteen, with 9.4 as the mean number of periods.

7 - Students were asked to indicate what the optimal number of students assigned to manage each bank would be. Responses ranged from one to four with a mean value of three.

8 - When asked to rate the contribution the formal project report made in the overall learning process, the mean ranking was 3.5 on an integer scale of one to five.

Finally, students were asked to provide additional comments or suggestions they felt would enhance the effectiveness of the game. A large number reiterated their desire to see the prime rate play a more realistic and integral part of local pricing decisions. One student suggested that students would find the game more challenging and rewarding if they were not given a copy of the 'Estimation of Investible Funds' schedule, but instead were required to develop such a decision-making tool as part of the simulation exercise itself.

Other suggestions included a request for additional economic data, more class time devoted to performance analysis on a period-by-period basis, investigating ways to make the simulation more exciting (such as an unexpected bank robbery or the discovery of embezzled funds), and one student suggested the game should be initiated earlier in the semester, with additional time alloted between individual decisions.

These responses serve to confirm student desire for simulation enhancements which would add greater realism to the learning exercise and provide added challenge by increasing the overall complexity of the game. It became apparent, in the student project reports, that it was possible for several of the teams to identify one or two key variables that critically influences bank performance. As one student phrased it, "During the course of the six-period game, our bank found it necessary to change its initial objectives after period three. This we feel was due to the fact, though the other two banks were profitable, that our competition was more concerned with "winning" the game. They failed to manage their banks in a manner that a prudent banker would, given the same economic data. This made our management team disagree on many subjects; whether to "play" the game and go for pure profit and ratios, or manage the bank in a manner which takes into account the rules and management principles we have learned during the semester."

Their suggestions regarding the prime rate, yield curve, sensitivity to pricing and expense decisions (advertising and office salaries), plus the introduction of credit risk via a loan loss account, reflects a desire for added realism and greater challenge. Expansion of the planning horizon from six to approximately ten periods, would presumably allow for substantially greater variation in expected interest rates and general economic conditions.

It was apparent that the students found it quite helpful to have been required to document their decision making process and then have the opportunity to formally describe and analyze the decision making process. This encouraged each team to clearly identify past mistakes and to explore alternative decisions that might have been adopted. Finally, the recognition that a formal project report was required and represented twenty-five percent of the student's final grade, provided added incentives to achieve maximum bank performance.

SECTION III

SURVEY RESULTS

In an attempt to evaluate OBS in a broader, more general perspective, a survey was made of those institutions which are currently using the model. The objective was to identify the various instructional environments within which the model is being used, and to assess the relative performance of the model in a variety of teaching applications.

A total of nine universities, in addition to Indiana University (Fort Wayne), are currently using or intend to use the model in the immediate future. Three distinct types of instructional applications were readily identified:

1 - Use in a course explicitly designed to teach the management and operating practices of financial institutions (similar to F-446).
2 - Use in an agricultural finance course to illustrate factors affectiving sources of agricultural credit and to provide management experience for potential agricultural loan officers.
3 - Use in a number of state banking schools oriented toward general bank management personnel or tailored to agricultural loan officers.

The various objectives, related operating information, and evaluation results associated with each application is summarized below.

Utilization in a Financial Institution Course

The Department of Marketing and Finance at Western Illinois University offers a course labeled Commercial Bank Management, and North Dakota State University, Department of Economics, utilizes OBS in a course entitled Money, Credit and Financial Markets. Detailed operating data related to its utilization is presented in Table II. A number of specific instructional objectives were identified by both institutions. In one case the model is being used to supplement course lectures while in the other course, the model serves to substitute for a series of lectures relating to applied bank management. In either case, OBS is being used to illustrate basic banking functions, such as investment portfolio management in a group decision making environment. One institution emphasizes the game theory aspect of the model with primary attention focused upon alternative competitive strategies.

Several suggestions were offered relative to desired changes in subsequent model revisions. It was pointed out that since no opportunity is currently provided for a bank to sell Federal Funds, the penalty for under-investment is unrealistically high. Adding the capability to sell Federal Funds is seen as providing a more effective and complete link to outside money markets. In addition, it was suggested that several modifications could be incorporated which would simplify administration of the simulation itself. An instructors diagnostic subroutine and an expanded instructors manual, which provides a general introduction to simulation theory, were among the specific suggestions offered. Caution was indicated relative to making the model too complex, suggesting that major modifications could be self-defeating, given the primary emphasis on fundamental management techniques.

TABLE II INSTRUCTIONAL PARAMETERS

	INSTITUTIONAL COURSE			AGRICULTURAL FINANCE COURSE			BANKING SCHOOLS			
	IU-PU Ft.Wayne	N.Dakota State	W. Ill. Univ.	Texas A&M	Ohio State	Okla. State	PURDUE	IOWA STATE	OKLA.ST UNIV.	TEXAS A&M
1. COURSE TITLE	Mgt.of Comrcl. Banks	Money, Credit & Fin. Mkts.	Comrcl. Bank Mgt.	Grad.Ag. Finance	Grad.Ag. Finance	Agricltr. Finance	2yr Bank Mgmt. & Ag.Lending Sch.	Sch. of Banking Grad.Ag School	Sch. of Banking Ag.Lend ing Sch	School of Bankng
2. HOURS OF CREDIT	3	3	4	3	3	3	-	-	-	-
3. REQUIRED OR ELECTIVE	R	E	E	E	E	R/E	-	-	-	-
4. # WEEKS IN SESSION	16	10	10	16	10	16	1	1-2	4 days	2 days
5. WKS. PER SESSION OF INSTRUCTION IN BANK PRINCIPLES	16	10	10	3	1	1	1	1-2	4 days	2 days
6. # WEEKS DEVOTED TO SIMULATION	4	4	4	5	6	5	1	1-2	4 days	2 days
7. AVERAGE # OF PARTICIPANTS IN CLASS	12	30	33	70	35	40	30	60	130	50
8. TYPICAL # ASSIGNED TO MANAGE SINGLE BANK	4	4	3	4	3	2	5	3-4	3	5
9. TOTAL # PERIODS OF PLAY	6	6	6	5	6	5	5	4-6	4	5
10. # TIMES/YEAR COURSE OFFERED	1	1	2	3	4	4	-	-	-	-
11. WERE INSTRUCTIONAL OBJECTIVES MET? (Min.= 1, Opt.= 5)	3	4	3.5	3	4	4	5	5	5	5
12. PERFORMANCE EVALUATION -Course Requirement -Grades Assigned	 YES YES	 YES NO	 YES NO	 YES NO	 YES NO	 YES NO	 YES NO	 YES NO	 YES NO	 YES NO
13. STUDENT RESPONSE (VE=Very Enthusiastic, ME=Moderatly Enthus., I=Indifferent)	ME	ME	ME	ME	ME	ME	VE	VE	VE	VE
14. % OF PARTICIPANTS WITH PREVIOUS SIMULATION EXPERIENCE	100%	30%	50%	10%	50%	50%	Unk.	Unk.	Unk.	Unk.
15. COMPUTER SYSTEM USED	CDC	IBM	IBM	AMDAHL	IBM	IBM	CDC	IBM	IBM	AMDAHL
16. INSTRUCTOR TIME (HRS)	23	12	40	4	12	15	-	10	-	-
17. PROGRAM MODIFICATION REQUIRED?	Extens.	Minimum	Extens.	Modest	Min.	Min.	-	Min.	Min.	Some
18. PLANS TO MODIFY?	YES	NO	NO	NO	NO	YES	NO	YES	YES	NO
19. PLAN TO RETAIN	YES	YES	YES	?	?	?	YES	YES	YES	YES
20. AVAILABLE TO LOCAL BANK GROUPS?	YES	YES	YES	IN PROCESS	YES	IN PROCESS	MAIN PURPOSE	MAIN PURPOSE	MAIN PURPOSE	MAIN PURPSE

Overall, the two institutions rated the degree to which their specific instructional objectives were achieved, 3.5 and 4.0 on a five point scale.

Utilization in an Agricultural Finance Course

OBS has been included in Agricultural Finance courses offered by the departments of Agricultural Economics at Ohio State University, Oklahoma State University, and Texas A & M University. Specific operating statistics are once again provided in Table II. Several of the objectives mentioned included:

1 - Providing the students with a general overview of the fundamental issues involved in commercial bank management.
2 - Illustrating some of the important constraints that commercial banks as important agricultural lenders face.
3 - Application of financial planning and asset management.
4 - Effectiveness in underscoring important aspects of bank competitions, providing an effective means of stimulating student participation and generating productive discussion.

One advantage identified was the minimum student effort and time required to play the game. This quality is of obvious value when time constraints seriously limit the extent of class participation. Each of the institutions stressed the importance of modifying Version Number Three to make the simulation more realistic and challenging.

Clint Rousch, at Oklahoma State University, indicates that he is presently working on Version Number Four, which introduces the capability of both buying and selling Federal Funds, with interest expense and interest receipts to be paid or received during the same period the transaction takes place. In addition, loan officer salaries will be directly related to improved bank performance.

Future modifications may involve the introduction of loan losses in some stochastic form, plus a re-estimation of the regression coefficients and response parameters to update the model. Another possible modification involved specification of a direct loan/deposit feedback relationship to realistically link the lending and deposit functions.

The model as presently constituted received a ranking of one 3 and two 4's, suggesting that the model was contributing to current instructional needs, while room for improvement is indicated. All of the institutions indicated some uncertainty as to whether or not the model would continue to be used as a supplement to the basic Agricultural Finance course. At Ohio State, the finance course is jointly taught by several faculty members, and no decision has yet been made regarding future use of OBS. At the other two universities, it appears that time constraints encountered in covering the more traditional finance material may necessitate droping the bank simulation exercise.

Utilization in a Banker's School

Purdue University, Texas A & M University, Oklahoma State University, and Iowa State University all are actively using OBS in a number of local banking schools. (See Table II for detailed operating information.) Typically, these schools are oriented towards new bank employees and middle management personnel, but it is not uncommon to have the president or senior staff members from a small rural institution attend the session. There appears to be two distinct groups effectively utilizing the model. One is a course of instruction or "school of banking" which frequently runs for two consecutive summers, with each summer session lasting one week. Here the primary emphasis is placed on general fundamentals of bank management. Another application is to introduce the OBS model in a more specialized course, designed to illustrate the principles associated with agricultural lending and to develop an appreciation of how agricultural lending compliments the remainder of the bank's loan portfolio.

For example, at Iowa State, OBS is being used to illustrate flow-of-funds concepts, investment portfolio analysis with primary emphasis upon the profitability impact of municipal securities, the overall effects of various exogenous economic variables on bank performance, plus the importance of interest rates and competitive bank behavior on market share and profitability performance. The bankers program in Oklahoma is currently in its fourth year with several hundred bankers having had the opportunity to participate in the simulation exercise. Typically, the bankers are solely responsible for their own bank's performance with little or no staff consultation. Performance is reviewed on a period-by-period basis, with an intensive review and wrap-up session at the end.

Each institution indicated that the simulation exercise was very well received by the large majority of the participating bankers. In at least one instance, the bankers inquired as to whether or not OBS could successfully operate on their own bank's computer system with little or no major modifications. It appears that system storage limitations for many small rural banks might prove to be an effective constraint in this regard. One possible alternative would be to offer the model on a continuing basis in the form of a correspondence course, supported by the University's computer facilities. In any event, OBS appears to be a highly successful tool for teaching and reinforcing basic bank management principles to bank personnel in a fairly intensive seminar training program.

SECTION IV

CONCLUSIONS AND RECOMMENDATIONS

The following conclusions and recommendations are derived from the authors' personal experience, using OBS in a formal class setting, the insights gained from a survey of current OBS users, plus an evaluation of alternative bank simulation models.

Conclusions

1. The simulation approach appears to be an effective technique for illustrating general principles of commercial bank management. Its primary utility stems from (1) the opportunity afforded students and bankers to actively participate in the management process, (2) the benefits associated with group discussion and close interaction, the need for compromise and team consensus, and (3) the opportunity to correlate specific bank performance to actual team decisions in a timely manner.

2. The current efforts to modify Version Three to make the simulation exercise more realistic and somewhat more challenging and complex should serve to enhance student interest and make it a more effective teaching tool. The possibility of explicitly introducing risk into the model in a formal manner should be

seriously considered.

3. A simulation model such as OBS ideally belongs in a management oriented course designed to bring previously acquired financial techniques to bear on basic problems facing commercial bank management. The model appears to be noticeably less successful when used to supplement a course primarily designed to teach basic financial principles and techniques.

4. Students participating in an academic simulation exercise should be encouraged to become as deeply involved with the "spirit" of the game as well as the specific techniques employed. Possible means of encouraging strong student interest include (1) explicitly evaluating team performance, (2) incorporating both team as well as individual performance into the students' final course grade in a meaningful way, (3) provide an element of "outside" competition into the gaming process, and (4) provide the students with the opportunity to formally develop meaningful objectives, realistic strategies, and helpful evaluation techniques, by encouraging them to express themselves in the form of a project summary or report.

Recommendations

1. Personal experience suggests that significant instructor experience in playing the simulation will assist one to be more effective in achieving maximum instructional impact with the model and provide a more effective means of illustrating specific management principles discussed in class. Thus, not only must the class become seriously involved, the instructor must be willing to devote significant class time plus out of class time to insure that all the implications of the various inter-relationships are fully recognized and understood by all the participants. A period-by-period joint instructor/student analysis of bank performance appears to be a useful means of accomplishing this objective. In addition, a final review session should be held to summarize overall student performance and to critique specific strategies and techniques employed during the simulation.

2. One means of stimulating increased student interest and providing some measure of optimal performance throughout the game, would be to include an outside or "ghost" management team composed of graduate students, interested faculty, or bank personnel. To promote additional competition and student interest, the authors would like to encourage the other Indiana University regional campuses as well as the Bloomington campus to give some careful thought to adopting OBS, with the idea of selecting a superior management team from each facility to meet at the conclusion of a semester for a systemwide simulation playoff. A substantially more complex alternative approach would be to have individual branch teams competing against one another on a continuous basis throughout the semester.

3. The simulation game could be used to model alternative managerial approaches presented in lecture. For example, the results of banks primarily concerned with market share goals versus income objective, or liability management versus fixed coefficient asset allocation could be set up and run outside of class with the output presented and discussed in lecture.

4. Given the high degree of success OBS has achieved when offered in a variety of banking seminars and trade association schools, the authors intend to make the model available to local bankers here in Fort Wayne, and in the surrounding communities. Several approaches are under consideration, but two likely methods would be to offer the simulation game in an AIB (American Institute of Banking) course, which typically receives widespread support from local bank personnel, or perhaps under the auspices of the newly established Center for Economic Education.

5. Given the number of bank simulation models currently available and their great diversity in terms of managerial comprehensiveness, technical complexity, input requirements, performance output, initial cost, and computer support requirements, it is advisable for each institution to carefully define its current as well as anticipated needs when electing to implement a simulation model. It may be possible to define a program which utilizes several alternative models to allow participating students and bankers the opportunity to "upgrade" their simulation capabilities as the situation warrants.

6. In order to keep the simulation as relevant and challenging as possible, various exogenous parameters and internal response coefficients should be changed and updated on a regular basis.

BIBLIOGRAPHY

Asher, Joe, "What Bank Schooling Can Be in the Computer Age," *Banking*, (Journal of American Bankers Association), August, 1975.

Boehlje, Michael, and Glen Fisher, "A Competitive Management Game for Non-Metropolitan Commercial Banks," *Journal of Bank Research*, Spring, 1976.

Boehlje, Michael, and Glen Fisher, "Educational Programs for Rural Bankers' Use of a Management Game," Economic Report No. 8, September, 1976, Iowa State University.

Fisher, Glen, Michael Boehlje and Clint Roush, "A Management Game for Non-Metropolitan Commercial Banks - A Game Manual," Research Report P-703, Agricultural Experiment Station, Oklahoma State University, Stillwater, October, 1974.

Green, J. R., and R. L. Sission, *Dynamic Management Decision Games*, John Wiley and Sons, New York, 1959.

Robichek, Alexander, Charles Haley and David Niebahr, "Stanford Bank Management Simulator," Graduate School of Business, Stanford University, 1972.

Robinson, Randall S., "BANKMOD: An Interactive Simulation Aid for Bank Financial Planning," *Journal of Bank Research*, Vol. 4, No. 3, Autumn, 1974, pp. 212-224.

Zuckerman, David W., and Robert F. Horn, *The Guide to Simulation Games for Education and Training*. Paul E. Twelker, Cambridge, Mass., 1970.

INTERACTIVE COMPUTER MODELS FOR INTRODUCING SIMULATION TECHNIQUES

WILLIAM C. NYLIN, JR. AND DAVID R. READ

LAMAR UNIVERSITY

Introduction

Interactive simulation models can be used as both motivational and instructional tools for the teaching of simulation techniques. The two models presented in this paper were designed with both concepts in mind. The models are a simulation of a shark chasing a girl and a population simulation.

It was the intent of each of these programs to meet the following objectives: (1) to be essentially self-explanatory during program execution, (2) to be sufficiently interesting and motivational that the students would play (work) with them, (3) to be simple enough and well documented enough to allow students to access and modify the programs for their own use, and (4) to demonstrate and teach problem solving using discrete time simulation techniques.

Each of the two programs were designed for use in introductory computer science and mathematics modeling courses at Lamar University. However, they were also widely used as demonstration programs for a variety of audiences. The shark program was especially popular with students. It was edited to produce a variety of different chase programs.

The simulations are run on a Honeywell 66/20 computer using interactive CRT terminals. The students are assumed to have little or no prior computer experience and are introduced to the BASIC programming language during the courses in which the models are used.

The Chase Simulation

The chase simulation is an example of a simple discrete time simulation model that is easy for an introductory computer science class to understand and code. As a result of using some of the excellent simulation examples in Introduction to Computer Science, by Rice and Rice [1] and FORTRAN for Humans, by Didday and Page [2] for many years in introductory computer classes, it was observed that these are highly motivational and a lot of fun for the students.

Again, as in the coin toss simulation, the program is developed in class. First a simple model where the shark chases a girl with velocities and distances input. This initial program only output the distances for each simulation period. Once the program is modified to plot the positions of the shark and the girl (very easy in BASIC), the program immediately becomes very interesting.

Permission to make fair use in teaching or research of all or part of this material is granted to individual readers and to non-profit libraries acting in their behalf, provided that CCUC's copyright notice is given and that reference is made to these Proceedings, their date of publication, and to the fact that reprinting privileges were granted by permission of the CCUC. To otherwise reprint a figure, table, other substantial excerpt, or the entire work requires specific permission as does republication, systematic or multiple reproduction.

© *1978 CCUC*

After this model is run in class several times, statements are inserted which have the relative velocities decay with time. The intent is to demonstrate how easily simulation models can be modified. The following output shows the shark's velocity decaying at 8% per second and the girl's at 3% per second. The plot allows the decays in the velocities to be easily seen.

```
RATES OF DECAY OF THE VELOCITIES OF SHARK, GIRL ( % / SEC) ? 8,3

WHAT ARE THE DIST, VEL OF SHARK AND DIST, VEL OF GIRL ? 160,15,80,5

BEACH                            GIRL                           SHARK
BEACH                          GIRL                        SHARK
BEACH                        GIRL                     SHARK
BEACH                      GIRL                SHARK
BEACH                     GIRL             SHARK
BEACH                   GIRL           SHARK
BEACH                  GIRL        SHARK
BEACH                GIRL        SHARK
BEACH               GIRL     SHARK
BEACH              GIRL   SHARK
BEACH            GIRL   SHARK
BEACH           GIRL SHARK
BEACH          GIRLSHARK
BEACH         GIRLSHARK
BEACH        GIRLSHARK
BEACH      GIRLSHARK
BEACH     GIRLSHARK
BEACH    GIRLSHARK
BEACH   GIRLSHARK
BEACH  GIRLSHARK
BEACH GIRLSHARK
BEACHGIRLSHARK
THE GIRL MADE IT TO SHORE IN 22 SECONDS.
```

Once the class has a good feel for this program, it is modified to allow for randomness. The previous example was completely deterministic. Each velocity is allowed to vary from 0 to twice the input velocity while keeping the input velocity as the average. A variable is introduced to scale the degree of randomness in the models, it can be seen that either the shark or the girl can easily win.

```
HOW RANDOM WOULD YOU LIKE THE VELOCITIES  (0 TO 100 %) ? 100

RATES OF DECAY OF THE VELOCITIES OF SHARK, GIRL (% / SEC) ? 8,3

WHAT ARE THE DIST, VEL OF SHARK AND DIST, VEL OF GIRL ? 160,15,80,5

BEACH                            GIRL                           SHARK
BEACH                          GIRL                 SHARK
BEACH                         GIRL               SHARK
BEACH                       GIRL          SHARK
BEACH                     GIRL     SHARK
BEACH                  GIRL       SHARK
BEACH                GIRL       SHARK
BEACH              GIRL      SHARK
BEACH              GIRL   SHARK
BEACH             GIRLSHARK
THE SHARK CAUGHT THE GIRL IN 10 SECONDS.
```

```
WHAT ARE THE DIST, VEL OF SHARK AND DIST, VEL OF GIRL ?160,15,80,5
BEACH                             GIRL                       SHARK
BEACH                          GIRL                        SHARK
BEACH                       GIRL                        SHARK
BEACH                    GIRL                        SHARK
BEACH                  GIRL                          SHARK
BEACH                GIRL                        SHARK
BEACH               GIRL                       SHARK
BEACH              GIRL                     SHARK
BEACH            GIRL                 SHARK
BEACH           GIRL            SHARK
BEACH        GIRL           SHARK
BEACH       GIRL           SHARK
BEACH       GIRL        SHARK
BEACH     GIRL       SHARK
BEACH    GIRL     SHARK
BEACH   GIRL    SHARK
BEACH  GIRL   SHARK
BEACH GIRL   SHARK
BEACHGIRL   SHARK
THE GIRL MADE IT TO SHORE IN 19 SECONDS.
```

Finally the model was modified to allow for any time increment. This allows the student to see the effect of a very fine simulation period versus a coarse simulation.

The Population Simulation

The population simulation was written to support the difference equations module of a new sophmore-level Introduction to Applied Mathematics Course at Lamar University. This low prerequisite (pre-calculus) course provides a requested service for Biology and Humanities majors, as well as being required for all Mathematical Sciences majors.

The diversity of backgrounds represented in each class presents quite a challenge in terms of: (1) providing meaningful examples which appeal to more than a small fraction of the class, (2) remaining within the constraints imposed by the fact that some of the students have not had a course in differential calculus (3) convincing the non-mathematics majors that mathematics is relevant to current problems in their fields, and (4) convincing the mathematics majors that there exist many interesting, challenging, and important problems amenable to mathematical treatment, which lie outside the areas traditionally associated with mathematics.

It was decided early in the development of the course that it should be very problem oriented. As far as the student is concerned, the problem will motivate the mathematics rather than vice versa. Problems are posed, and then the mathematical techniques are developed to solve them. In this manner the student is exposed to the practice of applied mathematics as it actually occurs in the "real world". Every effort is made to see that the student participates fully in each phase of the learning process.

The population simulation lends itself admirably to the solution of the above difficulties. Population problems are easily stated, have wide application, and the mathematical tools involved (in the discrete models) can be understood by students with only a pre-calculus background.

There are two basic population models (e.g. see [3],[4]) which can be run using this program:

(1) The standard linear growth model
$P(t+\Delta t) - P(t) = a\Delta t\, P(t)$

(2) The logistic model
$P(t+\Delta t) - P(t) = \Delta t\, P(t)\ (a - bP(t))$

Each model can also be run with a time delay:

(3) $P(t+\Delta t) - P(t) = a\Delta t\, P(t-\Delta t)$

(4) $P(t+\Delta t) - P(t) = \Delta t P(t)\ (a - bP(t-\Delta t))$

In addition, each of the above models can also be run with constant migration occuring. The program can be easily modified to allow time dependent migration. Each of the eight models (linear growth or logistic, with or without time delay, with or without migration) can also be run allowing the parameters to vary randomly about their mean values.

The following is an example of a current relevant problem which should appeal to students of biology, oceanography, government, etc., and which can be examined using the harvesting (migration) option of the population simulation program:

Since only 1000 to 1500 bowhead whales are believed to be in existance, the International Whaling Commission (IWC) has set a quota limiting the bowhead kill by American Eskimos to 12 per year. For centuries the bowhead has been a staple of the eskimo diet so the quota has not been well received. If the quota is disregarded, and the growth rate of the bowhead is 50 per thousand, per year, then how may whales can the eskimos kill per year without insuring the extinction of the species with 25 years?

The problem was solved by arbitrarily choosing kill rates of 10 per year and 100 per year (ie. migration rates -10 and -100). The resulting whale populations at the end of 25 years are 4602 and 306 respectively. Linearly extrapolating these results to estimate the kill rate, X, giving extinction at the 25 year gives:

$$X = 100 + \frac{10 - 100}{4602 - 306}\ (0 - 306)$$
$$= 106.4$$

The following runs indicate that a kill rate of more than 106 per year will eliminate the species in no more than 25 years.

```
DO YOU WANT TO RUN THE RANDOM MODEL?NO

OPTION 1 IS THE LINEAR GROWTH MODEL.  OPTION 2 IS THE LOGISTIC
MODEL.  WHICH OPTION DO YOU CHOOSE?1

WHAT ARE THE: INITIAL POPULATION AND INTRINSIC GROWTH RATE?1500,.05

WHAT IS THE TIME PERIOD AND HOW MANY DO YOU WISH TO RUN
I.E. WHAT ARE DELTA T, AND N?1,25

DO YOU WANT A TIME DELAY (YES OR NO)?NO

DO YOU WANT MIGRATION TO OCCUR?YES
```

```
ENTER NUMBER MIGRATING PER UNIT TIME (PLUS OR MINUS)?-106

 TIME AND POPULATION ARE:

                 0            1500
                 1            1468
                 2            1436
                 3            1402
                 4            1366
                 5            1328
                 6            1289
                 7            1247
                 8            1203
                 9            1158
                 10           1110
                 11           1059
                 12           1006
                 13            950
                 14            892
                 15            831
                 16            766
                 17            698
                 18            627
                 19            553
                 20            474
                 21            392
                 22            306
                 23            215
                 24            120
                 25             20

  DO YOU WISH TO RUN THE MODEL AGAIN?YES

DO YOU WANT TO RUN THE RANDOM MODEL?NO

OPTION 1 IS THE LINEAR GROWTH MODEL.  OPTION 2 IS THE LOGISTIC
MODEL.  WHICH OPTION DO YOU CHOOSE?1

WHAT ARE THE:  INITIAL POPULATION AND INTRINSIC GROWTH RATE?1500,.05

WHAT IS THE TIME PERIOD AND HOW MANY DO YOU WISH TO RUN
I.E. WHAT ARE DELTA T, AND N?1,25

DO YOU WANT A TIME DELAY (YES OR NO)?NO

DO YOU WANT MIGRATION TO OCCUR?YES
```

```
ENTER NUMBER MIGRATING PER UNIT TIME (PLUS OR MINUS)? -107

 TIME AND POPULATION ARE:

              0           1500
              1           1467
              2           1434
              3           1399
              4           1362
              5           1323
              6           1282
              7           1239
              8           1194
              9           1147
             10           1097
             11           1045
             12            990
             13            933
             14            872
             15            809
             16            742
             17            673
             18            599
             19            522
             20            441
             21            356
             22            267
             23            174
             24             75
             25            -28

          THE POPULATION HAS BECOME EXTINCT
```

Thus a kill rate of more than 106 per year will eliminate the species in less than 25 years.

Summary

These simulations have proven quite successful in the classroom. Students are very enthusiastic about the interactive terminals, and most of them are not only willing, but glad to spend several hours per week in the terminal room. The time spent in (math) class teaching the BASIC programming language and program access and execution was returned with interest in terms of the student enthusiasm, variety of, otherwise unfeasible (due to time constraints, complexity of collecting data, etc.) problems which can be considered, and the pedagogically sould principle of actively involving as many of the students' senses as possible in the learning process.

The ability to access and modify these programs outside of class was extremely successful, Since students can easily modify programs, the chase program rapidly appeared in many forms in the students disk files.

After seeing these models, the students were able to design and implement their own simple simulation programs with a minimum of difficulty.

References

1. J.K. Rice, and J.R. Rice, Introduction to Computer Science, Holt, Rhinehart and Winston, Inc., 1969.
2. Didday, and Page, Fortran for Humans, West Publishing Co., Inc., 1974.
3. Richard Haberman, Mathematical Models Mechanical Vibrations, Population Dynamics, and Traffic Flow, Prentice-Hall, Inc., 1977.
4. S.I. Grossman, and J.E. Turner, Mathematics for Biological Sciences, MacMilliam Publishing Co., Inc., 1974.